HOCHSCHULE
MITTWEIDA
UNIVERSITY OF
APPLIED SCIENCES

DIPLOMARBEIT

Christoph Köhle

Steuergerät zur Einstellung der Kameraplattform T^2Control

Mittweida, 2011

Fakultät Elektro- und Informationstechnik

DIPLOMARBEIT

Steuergerät zur Einstellung der Kameraplattform T^2Control

Autor:
Ing. Christoph Köhle

Studiengang:
Informationstechnik

Seminargruppe:
KI09wIA

Erstprüfer:
Prof. Dr.-Ing. Thomas Beierlein

Zweitprüfer:
Dipl.-Ing. (FH) Christian Neuner

Einreichung:
Mittweida, 31.10.2011

Verteidigung/Bewertung:
Mittweida, 2011

Bibliografische Beschreibung:

Christoph Köhle:

Entwicklung eines Eingebetteten Systems (Hard- und Software) **T²Control** (Turn Table Control) zur Fernsteuerung einer horizontal und vertikal drehbaren Kameraplattform an einer Flugdrohne für technische Vermessung, Photogrammetrie und Fernerkundung. – 2011 – 84 Seiten.

Hochschule Mittweida, Fakultät Elektro- und Informationstechnik, Diplomarbeit, 2011

Referat:

Die vorliegende Arbeit befasst sich mit der Entwicklung eines Steuergerätes als eingebettetes Systems, zur Einstellung der Kameraposition des VTOP (Vertikal Take Off Platform) über Funk.

Nach der Beschreibung der verwendeten Hardwarekomponenten wird die Prototyp – Hard – und Software entwickelt und dokumentiert. Es folgen, Hard- und Softwaretests, Implementierung der Software und Integration des Steuergeräts in das bestehende System.

Eine Zusammenfassung der Ergebnisse sowie ein Ausblick auf mögliche Optimierungen schließen diese Arbeit ab.

Inhalt

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

CPU	**C**entral **P**rocessing **U**nit
CSMA	**C**arrier **S**ense **M**ultiple **A**ccess
EEPROM	**E**lectrically **E**rasable **P**rogrammable **R**ead-**O**nly **M**emory
FSK	**F**requency **S**hift **K**eying
GPS	**G**lobal **P**ositioning **S**ystem
ISM	**I**ndustrial, **S**cientific and **M**edical Band
ISP	**I**n-**S**ystem-**P**rogrammer
LCD	**L**iquid **C**rystal **D**isplay
LED	**L**ight **E**mitting **D**iode
MAC	**M**edia **A**ccess **C**ontrol
NeCON	**Ne**uner **Con**sulting
RAM	**R**andom-**A**ccess-**M**emory **(**Speicher mit wahlfreiem/direktem Zugriff)
SRAM	**S**tatic **R**andom-**A**ccess **M**emory
UART	**U**niversal **A**synchronous **R**eceiver **T**ransmitter
VTOP	**V**ertical **T**ake **O**ff **P**latform

1 Übersicht

Im einleitenden Kapitel werden die Motivation und die Aufgabenstellung dieser Diplomarbeit besprochen. Gleichzeitig erfolgt ein kurzer Überblick zu den einzelnen Kapiteln dieser Arbeit.

1.1 Motivation

Die Firma NeCON – Neuner Consulting wurde im Jahre 2001 gegründet und bietet Dienstleistungen in folgenden Bereichen an:

- Technische Vermessung
- Photogrammetrie und Fernerkundung[1]
- Umweltmesstechnik

Die Firma NeCON ist im nationalen und internationalen Vergleich mit ihren 5 Mitarbeitern sehr klein, ist aber aufgrund ihrer schlanken Struktur in der Lage speziell in Nischenbereichen flexible und anwendungsorientierte Lösungen anzubieten.

Die Anfrage eines Architekten, ob es möglich wäre ein Panoramabild aus einer Höhe von 70 bis 80 Meter anzufertigen, veranlasste die Fa. NeCON sich mit dem im folgenden Abschnitt beschriebenen Projekt zu beschäftigen. Das Architekturbüro wollte für ein in Planung befindliches Bürogebäude eine Panoramaaufnahme in 80m Höhe erstellen lassen, um den Auftraggebern den Ausblick aus dem zukünftigen Büro zu zeigen. Dieser Auftrag wurde mit einer ferngesteuerten Flugdrohne realisiert, die sich im Besitz der Firma befand. An dieser war eine digitale Spiegelreflexkamera fix montiert.

Nachdem das Ergebnis zwar für den Auftrag ausreichend war, aber es in näherer Umgebung eine kleine Firma gab, welche mit ähnlichem System bereits Dienstleistungen wie Foto- und Filmaufnahmen für Werbezwecke anbot, beschloss man ein professionelleres System, bestehend aus einem Systemträger (Drohne) und einer

[1]Photogrammetrie (seltener auch Fotogrammetrie oder Bildmessung) ist eine Gruppe von Messmethoden und Auswerteverfahren der Fernerkundung, um aus Fotografien und genauen Messbildern eines Objektes seine räumliche Lage oder dreidimensionale Form zu bestimmen. Im Regelfall werden die Bilder mit speziellen Messkameras aufgenommen. (Wikipedia unter Suchbegriff „Photogrammetrie“ verfügbar am 14.07.11)

sphärisch einstellbaren Kameraplattform in 2 Achsen für folgende Einsatzgebiete zu entwickeln:

- 2D bzw. 3D Bildausarbeitung und Digitalisierung
- Panoramabilder für technische Vermessung und Visualisierung aus der Bauträgerbranche

Die neue Flugdrohne sollte wesentlich größer sein, dass diese mit einer Spiegelreflexkamera bzw. mit Systemkameras auf einer drehbaren Plattform ausgestattet werden kann. Beim alten System kam es durch das Drehen des Fluggerätes in der Luft zu einem großen Höhenversatz, infolge der Lageregelung, der eine vermessungstechnische Auswertung der Bilder nicht möglich machte.

1.2 Zielsetzung

Aus der im vorangegangenen Abschnitt erklärten Motivation entstand ein Fluggerät, dessen Eigenschaften im Kapitel 2 näher erörtert werden.

Abbildung 1-1: Flugdrohne VTOP

Die Steuerung der Kameraplattform und der Kamera erfolgt derzeit mit einem Terminalprogramm und soll durch ein Steuergerät in Form eines Handheld-Gerätes ersetzt werden.

Für die Umsetzung der Arbeit sind neben der theoretischen Ausarbeitung eine Produktion eines Prototyps und die Erstellung der Firmware geplant. Die Entwicklung des Prototyps hat unter Berücksichtigung einer Kleinserienfertigung zu erfolgen. Die Softwareentwicklung wird mit der Hochsprache Bascom der Firma MCS Electronic realisiert. Diese Auswahl wird durch die Fa. NeCON vorgegeben.

Als Abschluss der Arbeit ist der praktische Betrieb des Steuergerätes im Feldeinsatz vorgesehen.

1.3 Kapitelübersicht

In diesem Abschnitt werden die Inhalte der einzelnen Kapitel kurz erläutert.

Kapitel 1:

Im einleitenden Kapitel werden die Motivation und die Aufgabenstellung dieser Diplomarbeit besprochen. Gleichzeitig erfolgt ein kurzer Überblick zu den einzelnen Kapiteln dieser Arbeit.

Kapitel 2:

Im Kapitel 2 wird der momentane Stand der Technik erläutert. Die Eigenschaften und Nachteile der derzeitigen Lösung werden aufgezeigt, um die Notwendigkeit der vorliegenden Arbeit zu rechtfertigen.

Kapitel 3:

Im Kapitel 3 werden die Ziele, welche mit der Arbeit erreicht werden sollen, definiert.

Kapitel 4:

Das Kapitel 4 beschreibt die Hardwareentwicklung. Die Arbeitsschritte von der Auswahl der Hardwarekomponenten bis zum fertigen Gerät werden erklärt.

Kapitel 5:

Das Kapitel 5 ist der Softwareentwicklung gewidmet und soll die Vorgehensweise bei der Programmerstellung verdeutlichen.

Kapitel 6:

In diesem Kapitel wird der Test von Hard- und Softwarekomponenten dokumentiert.

Kapitel 7:

Im abschließenden Kapitel werden die bisher gewonnenen Ergebnisse zusammengefasst und eine Bewertung der Leistung aus Sicht des Autors vorgenommen. Ein Ausblick zeigt Weiterentwicklungspotenziale auf.

2 Stand der Technik

In diesem Kapitel soll die derzeitige Lösung mit deren Eigenschaften erläutert werden.

2.1 Darstellung der aktuellen Lösung mit Ihren Eigenschaften

- Der bereits vorhandene Hexakopter[2] wird mit der Kameraplattform (siehe Abbildung 2-2) gekoppelt. Der Hexakopter ist eine universelle Schwebeplattform, welche in der Lage ist senkrecht zu starten. Er kann über eine Funkfernsteuerung gesteuert werden, ist aber auch mit GPS[3] ausgestattet und kann automatisch eine Position anfliegen und diese halten, oder eine vorgegebene Route abfliegen und selbständig zum Startpunkt zurückkehren. Der Hexakopter wurde von der Fa. NeCon entwickelt und gebaut. Es gibt zwar am Markt einige Anbieter für solche Systeme, diese sind aber nicht in der Lage die geforderte Nutzlast bestehend aus der drehbaren Kameraplattform mit montierter Spiegelreflex- oder Systemkamera, zu tragen. Die angebotenen Drohnen (siehe zum Beispiel Literaturquelle /17/) sind oft sogar mit einer Kamerahalterung ausgestattet. Diese sind aber nicht horizontal drehbar. Die angeführten Gründe rechtfertigten die Entwicklung eines Gesamtsystems, bei dem die Drohne mit der drehbaren Kameraplattform kompatibel ist.

[2] Hexakoper: Flugdrohne mit 6 Rotoren

[3] GPS (Global Positioning System): GPS basiert auf Satelliten, die mit kodierten Radiosignalen ständig ihre aktuelle Position und die genaue Uhrzeit ausstrahlen. Aus den Signallaufzeiten können spezielle GPS-Empfänger dann ihre eigene Position und Geschwindigkeit berechnen.

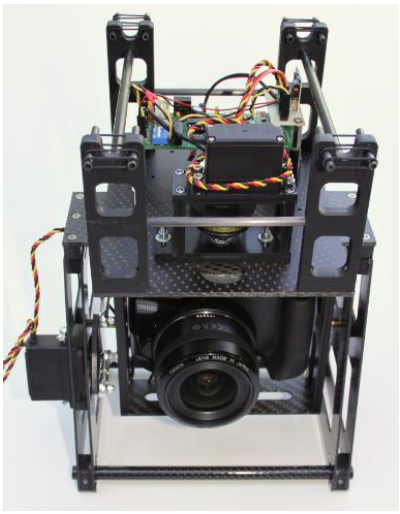


Abbildung 2-2: Kameraplattform mit montierter Kamera

- Die Kameraplattform ist als eine vom Hexakopter unabhängige Komponente ausgeführt und wurde für den universellen Einsatz auch ohne Flugdrohne entwickelt. In Verbindung mit der Flugdrohne sind aber die zur Energieversorgung der Plattform dienenden Akkus in der Drohne untergebracht. Das ist notwendig um eine optimale Gewichtsverteilung für den Flugbetrieb zu erreichen. Die Kameraplattform ist mit speziellen Gummiringen am Hexakopter montiert und kann mit wenigen Handgriffen demontiert werden. Somit ist auch die Verwendung auf einem Stativ für Panoramaaufnahmen oder Vermessungen vom Boden aus möglich. Die Ausrichtung des Objektivs der Kamera im Raum erfolgt über zwei drehbare Achsen der Kameraplattform. Die Rotation der Achsen wird mit Servomotoren durchgeführt. Die Positionierungsdaten zur Ansteuerung der Elektronik für die Servomotoren werden über Funk von einem Terminalprogramm der Bodenstation an eine Empfangseinheit übertragen. Die Empfangseinheit besteht aus einem Funkmodul, einer Auswerte- und Steuerelektronik. Für die Kommunikation mit der Bodenstation dient das Funkmodul Wi.232, welches später noch genauer erläutert wird. Die Empfangselektronik der Kameraplattform wertet die von der Bodenstation gesendeten Kommandos (z.B. Horizontal – oder Vertikalposition) aus und steuert die entsprechenden Servomotoren an. Die aktuelle Position der Kamera wird ausgewertet und der Bodenstation eine Statusmeldung übermittelt. Das Auslösen der Kamera erfolgt ebenfalls über ein Kommando von der Bodenstation.

- Die Bodenstation besteht derzeit aus einem Notebook mit einem selbstgebauten Sendegerät der Fa. NeCon. Das Sendegerät (siehe Abbildung 2-3) besteht im Wesentlichen aus dem Funkmodul Wi.232 mit einer Pegelumsetzung (MAX232). Das Funkmodul ist mittels RS232-USB-Konverter mit dem Notebook verbunden, auf dem ein Terminalprogramm läuft.

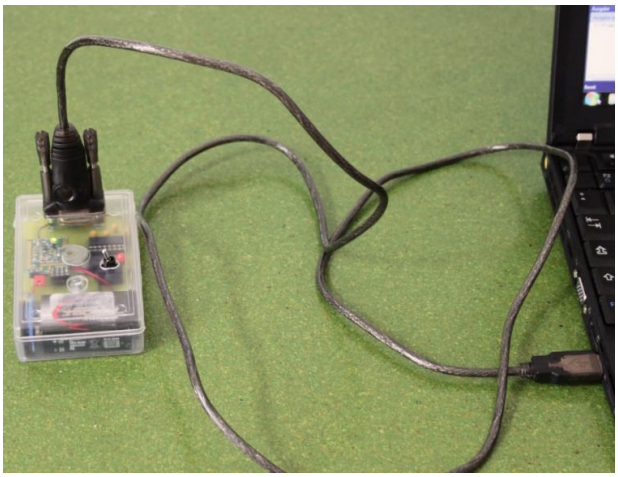


Abbildung 2-3: Sendegerät der derzeitigen Lösung

- Bei dem Terminalprogramm handelt es sich um ein frei erhältliches Programm mit der Bezeichnung Tera Term Version 4.68 (vgl./12/).
  Das Terminalprogramm dient dazu, die Tastatureingaben an die serielle Schnittstelle zu senden bzw. empfangene Daten am Bildschirm darzustellen. Mit der Tastatur des Notebooks wird die gewünschte Position eingegeben. Nach Empfang der aktuellen Kameraposition, welche am Terminal angezeigt wird, kann die Kamera mittels Tastatureingabe ausgelöst werden. Dann wird die nächste Position eingegeben und so weiter, bis die notwendigen Aufnahmen gemacht worden sind.

2.2 Bewertung der aktuellen Lösung

Die derzeitige Lösung mit der Steuerung über das Terminalprogramm erwies sich als nicht tauglich für den Feldeinsatz aus mehreren Gründen:

1. Bei schönem Wetter ist der Notebookbildschirm wenn überhaupt nur sehr schwer ablesbar.
2. Die Befehle über die Tastatur einzugeben ist zeitaufwendig, umständlich und somit nicht ergonomisch.
3. Hoher Geräteaufwand mit Notebook und externer Funkübertragungseinheit;
4. Hoher Zeitaufwand bis das System betriebsbereit ist;

5. Eine Vermessung durchzuführen ist derzeit nur mit zwei Personen möglich.

Ziel dieser Diplomarbeit ist es die angeführten Nachteile zu beseitigen und das beschriebene System für die gestellten Anforderungen im Praxiseinsatz tauglich zu machen.

2.3 Beschreibung der zu verwendenden Hauptkomponenten

Der Mikrocontroller ATmega32

Der ATmega 32 entstammt der AVR – Mikrocontroller – Familie von Atmel. Es handelt sich um einen 8 Bit Controller mit RISC[4] - Architektur. Die verschiedenen Typen der AVR – Familie unterscheiden sich voneinander durch die Größe des Programmspeichers (Flash), Größe des Datenspeichers (EEPROM), Größe des internen Datenspeichers (SRAM), der Anzahl der Ports und Analog-Digital-Converter, UART, Timer und dergleichen.

(Die Daten wurden der Literaturquelle /7/ entnommen.)

Der ATmega32 im Überblick:

- 8 Bit Mikrocontroller
- Maximale Taktfrequenz 16 MHz
- 32 kByte Flash
- 1 kByte EEPROM
- 2 kByte SRAM
- 2x8 Bit Timer/Counter
- 1x16 Bit Timer /Counter
- 8x10 Bit A/D Wandler
- UART

[4] Reduced Instruction Set Computer (RISC) (engl. für *Rechner mit reduziertem Befehlssatz*)

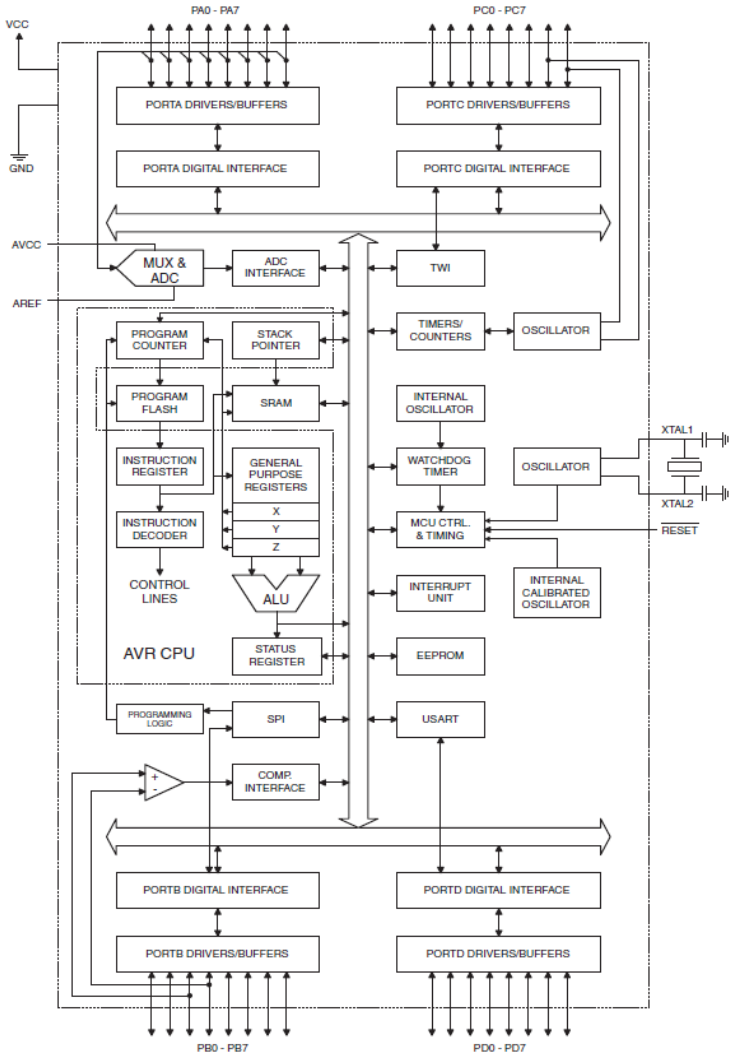


Abbildung 2-4: Blockschaltbild des ATmega32
(Quelle:/7/)

AVR CPU – Kern:

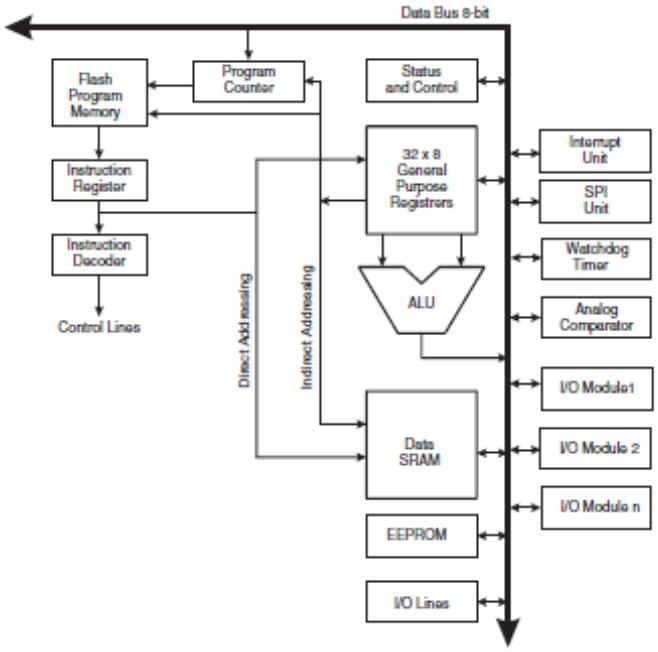


Abbildung 2-5: Blockschaltung Analog/Digital - Converter
(Quelle:/7/)

Der Mikrocontroller verfügt über eine Harvard – Architektur. Das heißt Programmspeicher und Arbeitsspeicher sind getrennt ausgeführt und über separate Busse an die CPU gekoppelt. Dadurch entsteht der Vorteil, dass Programm- und Dateninformationen zur gleichen Zeit über die getrennten Busse übertragen werden können.

Funkübertragungsmodul:

Die Funkübertragung erfolgt im ISM-Band[5].

Zur Datenübertragung zwischen dem Steuergerät und der Kameraplattform wird, wie bereits erwähnt, das Funkmodul Wi.232 verwendet. Die Sendefrequenz des Moduls beträgt 868 MHz. Das Funkmodul wird im folgenden Abschnitt genauer erklärt.

[5] **I**ndustrial, **S**cientific and **M**edical Band

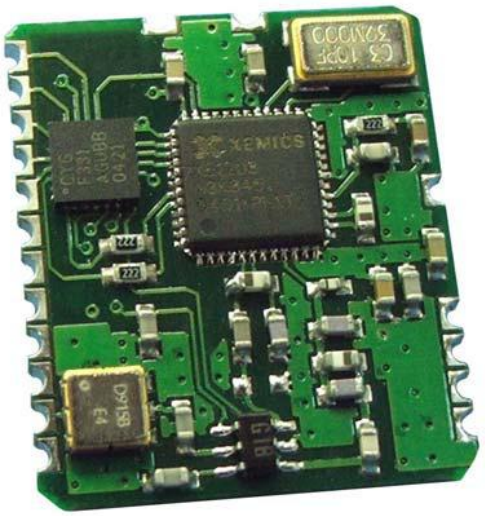


Abbildung 2-6: Funkmodul WI.232
(Bildquelle /10/)

Funktionsweise des Wi.232-Modules:
(Folgende Beschreibung wurde sinngemäß der Literaturquelle /10/ entnommen.)

Das Wi.232 Modul gehört zur Famile der WiSE™ (Wireless Serial Engine) Module. Ein WiSE™ Modul kombiniert einen State-of-the-Art Wideband/FSK Data Transceiver und einen High-Performance Protocol Controller.

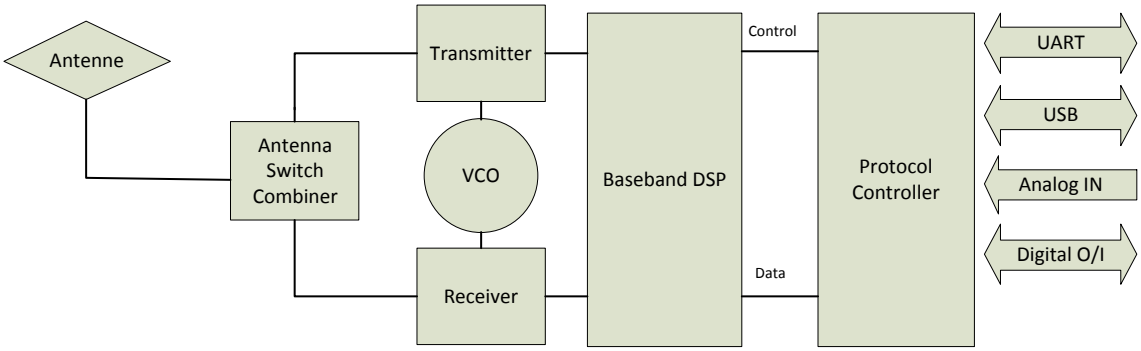


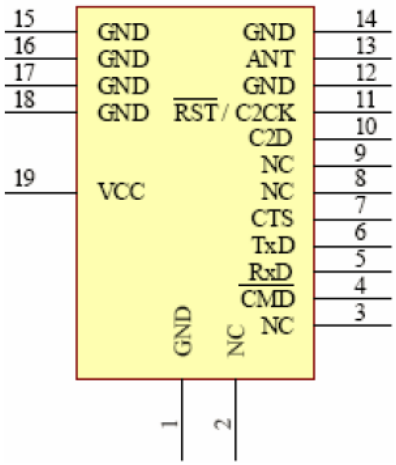
Abbildung 2-7: Blockschaltbild des WI.232 EUR

Das Wi.232EUR Modul besitzt ein UART-Type Serial Interface und enthält eine spezielle Systemsoftware um eine UART- to -Antenna- Wireless Lösung zu ermöglichen. Das Modul kann als Kabelersatz in Embedded RS-232/422/485 Applikationen verwendet werden.
Das heißt, überall wo eine serielle Kabelverbindung benötigt wird, kann das Wi.232 Modul diese Verbindung über Funk herstellen. Die Anwendung merkt nicht ob die Verbindung über ein Kabel oder drahtlos erfolgt.
Es sind *Point to Point* oder *Point to Multipoint*-Verbindungen möglich.

Die Module können in Gruppen arbeiten. Jedem Modul kann eine 7-Bit Gruppen-ID zugeordnet werden. So ist es möglich, dass alle Module, die auf dem gleichen Kanal arbeiten, untereinander kommunizieren können.
Die Module können auch in zwei Network-Modes betrieben werden: *Master/Slave* und *Peer-to-Peer*. In diesem Modus definiert ein Set von Kommunikationsregeln, welche Module untereinander kommunizieren dürfen.
Im *Master/Slave - Mode* kann der Master Verbindung mit Slaves oder anderen Mastern aufnehmen. Slaves können Verbindung mit anderen Mastern, aber nicht zu Slaves aufnehmen. Dieser Modus wird in RS-485 Netzwerken eingesetzt.
Im *Peer-to-Peer Mode* kann jedes Modul Verbindung zu jedem Modul herstellen.
In beiden Modes muss aber die gleiche Gruppen-ID eingestellt werden.
Wenn ein Modul ein Paket sendet, empfangen alle Module auf diesem Kanal das Signal, prüfen das Paket auf Fehler und überprüfen ob die Gruppen-ID passend ist. Wenn das Paket fehlerfrei ist und die Gruppen-ID passt, entschlüsselt das Modul das Paket und sendet die fehlerfreien Daten zum UART.
Die meisten Features des Moduls sind über programmierbare Register einstellbar. Die Register lassen sich lesen und beschreiben wenn der Command-Pin (CMD) Low ist.
Wenn CMD auf High liegt, werden alle Daten transparent über Funk gesendet.
Die Module enthalten zwei Kopien der Registerwerte: Im Flash und im RAM.
Bei einem Reset wird der Inhalt des Flashspeichers ins RAM geladen. Die Module arbeiten immer mit dem Inhalt des RAM Registers. Programme welche die Register des Moduls oft ändern, sollten den RAM Inhalt ändern.
Durch das Ablegen der Default Einstellungen im Flash, starten die Module immer mit gültigen Werten. Das ist für Anwendungen wichtig, die einfach nur Daten übertragen wollen aber nicht in der Lage sind die Registerinhalte zu verändern.
Das UART Interface überträgt in Full- Duplex mit Baudraten von 2,4 kbit/s bis 115,2 kbit/s.

Das Modul hat 10 Power- Modes: 4 Wideband- Modes, 4 Narrowband -Modes, Standby, und Sleep.
Im Wideband-Mode hat das Modul eine Bandbreite von 600kHz. In diesem Modus

kann es auf 2 Kanälen arbeiten und bringt es auf eine maximale Datenrate von 76,8kbit/s.
Im Narrowband- Mode ist die Bandbreite auf 200kHz begrenzt. In diesem Modus kann das Modul auf 6 Kanälen arbeiten und unterstützt bis zu 9600 Baud.
Das Modul kann per Kommando in einen Sleep-Modus versetzt werden, der HF-Teil wird ausgeschaltet und der Prozessor in einen Wartezustand gesetzt, die Stromaufnahme geht dann gegen Null.
Wenn der MAC Layer ein Paket senden möchte, benutzt er das Carrier-Sense-Multiple-Access (CSMA) Protocol um festzustellen ob ein anderes Modul gerade sendet. Sollte ein anderes Modul senden, wird diese Sendung empfangen und das Paket zu einem späteren Zeitpunkt gesendet.



No.	Description
1	Ground
2	No connect – reserved
3	No connect – reserved
4	Command input – active low
5	UART receive input
6	UART transmit output
7	UART clear to send output – active low
8	No connect – reserved
9	No connect – reserved
10	Reserved – ISP pin
11	Active-low Reset / Reserved – ISP pin
12	Ground
13	Antenna port – 50 ohm
14	Ground
15	Ground
16	Ground
17	Ground
18	Ground
19	VCC – 2.7 to 3.6 VDC

Abbildung 2-8: Pin – Belegung des WI.232
(Quelle /10/)

LC – Display

Dot-Matrix LC-Displays sind aufgrund ihres kleinen Preises und des geringen Aufwandes zur Ansteuerung in sehr vielen Mikrocontroller- gesteuerten Systemen als Ausgabeeinheiten in Verwendung. Im folgenden Abschnitt wird auf die Ansteuerung und auf die Funktion solcher Displays etwas näher eingegangen.

Typen von Text - Displays:

LC-Displays werden von vielen Herstellern weltweit angeboten. Die Anzahl der darstellbaren Zeichen ist sehr unterschiedlich. Es gibt einzeilige, zweizeilige und vierzeilige Typen, die jeweils 8, 16, 20 oder 40 Zeichen je Zeile darstellen können. Jedes Zeichen besteht aus einer Matrix aus 5 x 8 Punkten. Die Ansteuerung dieser unterschiedlichen Typen ist aber einheitlich und soll im weiteren Abschnitt näher beschrieben werden.

Beschaltung von Displays:

Die folgenden Anschlussbelegungen gelten für die meisten Displays. Es gibt aber auch Ausnahmen, deshalb ist es ratsam im Zweifelsfall das Datenblatt des Displays zu studieren.

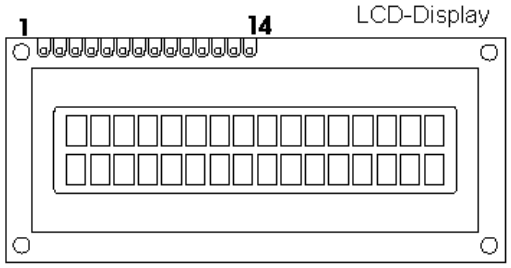


Abbildung 2-9: LCD Pin – Anordnung
(Quelle /11/)

Die 1- und 2-Zeiligen Displays (sowie 4-zeilige Displays mit bis zu 20 Zeichen pro Zeile) besitzen eine Reihe von 14 oder 16 Lötkontakten, bzw. eine einreihige Stiftleiste mit 14 oder 16 Kontakten. Die Kontakte 15 und 16 sind für den Anschluss der Hintergrundbeleuchtung (soweit vorhanden). Hat ein hintergrundbeleuchtetes Display nur 14 Kontakte am Verbinder, so sind die Beleuchtungsanschlüsse an einer anderen Stelle des Displays zu finden. Die Pins 1 bis 14 sind in aller Regel identisch belegt.

Manchmal findet sich auch ein zweireihiger 14- oder 16-poliger Steckverbinder. Die Nummerierung der Pins ist in der Regel auf der Platine angegeben. Die Funktionsbelegung der Pins ist identisch mit der Standardbelegung des einreihigen Verbinders.

LC-Displays besitzen folgende Anschlusspins:

- *Spannungsversorgung (Vdd und Vss, Pins 1 und 2):* Die LC-Displays werden mit einer Betriebsspannung Vdd von +5V betrieben. Vss ist der Masseanschluss. Die Stromaufnahme beträgt ohne Beleuchtung meist unter 1mA, höchstens aber 5mA.
- *Spannung zur Kontrasteinstellung (Vo, Pin 3):* Die Kontrasteinstellung wird mit einem 10kΩ-Potentiometer realisiert.
- *8-Bit-Datenbus (D0 bis D7, Pin 7 bis 14):* Dieser Datenbus dient zur Übertragung der Daten zum/vom Display (schreiben oder lesen) und kann entweder 8 Bit oder nur 4 Bit breit erfolgen. Der Vorteil des 4- Bit-Mode gegenüber dem 8-Bit-Mode ist die geringere Anzahl an Anschlusspins zum Mikrocontroller. Dieser Vorteil wird aber durch eine etwas umfangreichere Software eingebüßt.
- *Enable - Leitung (E, Pin 6):* Mit diesem Anschluss kann das Display deaktiviert bzw. aktiviert werden. Nur wenn die Enable-Leitung auf High-Pegel liegt, kann das Display angesprochen werden.
- *Read/Write-Leitung (R/W, Pin 5):* Dieser Anschluss bestimmt, ob Daten zum Display geschrieben (R/W = 0) oder, ob Daten vom Display ausgelesen werden (R/W = 1).
- *Register-Select-Leitung (RS, Pin 4):* Dieser Anschluss bestimmt, ob es sich bei den zum Display geschriebenen Daten um Steuerbefehle für das Display (RS = 0), oder um am Display darstellbare Zeichen (RS = 1) handelt.
- *Optional: Hintergrundbeleuchtung (A und K, Pin 15 und 16):* Sofern diese beiden Anschlüsse vorhanden sind, kann damit das Display beleuchtet werden. Dabei muss der Abschluss A (Anode) mit der Betriebsspannung (+5V) und der Anschluss K (Kathode) mit der Masse verbunden werden.

Der Controller HD44780:

Dass alle LC-Displays einheitlich angesteuert werden liegt daran, dass alle den Hitachi-Controller HD44780 eingebaut haben. Dieser hat sich als Standard durchgesetzt und wird von allen LC-Display-Herstellern verwendet. Es gibt aber noch Controller anderer Hersteller, die aber kompatibel zum HD44780 sind.

3 Präzisierung der Aufgabenstellung

In diesem Kapitel erfolgt die Präzisierung der Aufgabenstellung.

3.1 Blockdiagramm Übersicht

Die Steuerung der Kameraplattform erfolgt derzeit wie in Abbildung 3-10 ersichtlich mittels Notebook und Terminalprogramm in Verbindung mit einem externen Funkmodul. Diese Anordnung erwies sich im praktischen Einsatz als nicht optimal, wie im Abschnitt 2 beschrieben.

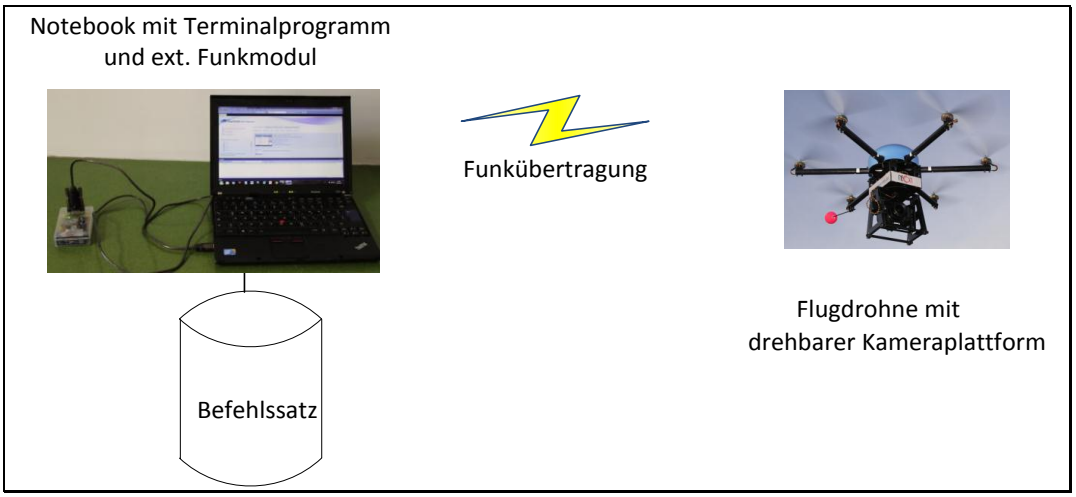


Abbildung 3-10: Übersicht Istzustand

3.2 Ziele der Arbeit

Die Anordnung Notebook mit externem Funkmodul soll nun mit einem zu entwickelnden Handsteuergerät ersetzt werden und so den Feldeinsatz in Zukunft wesentlich vereinfachen.

Das Steuergerät soll folgende funktionale Kriterien erfüllen:

- Manuelles Anfahren jedes beliebigen Winkels der Kamera in horizontaler und vertikaler Richtung

- Automatisches Anfahren von Winkelpositionen. Dabei sollen die verschiedenen Winkel-Schritte mit Ablaufprogrammen vorgegeben werden.
- Auslösen der Kamera.
- Kalibrierung der Null-/Maximalpositionen in der Azimut- und Elevationsebene.
- Interaktion mit dem Benutzer über Punktmatrix – Display und Tastatur

Anmerkungen zum Design des zu entwickelnden Steuergerätes:

- Die gesamte Elektronik soll in einem ergonomischen Handheld-Gehäuse untergebracht werden.
- Datenübertragung zur Servo-Elektronik via Funk im ISM-Band
- User-Interface mit Folientastatur und Punkt-Matrixanzeige
- Manuelle Einstellung der Winkel mit Schiebereglern

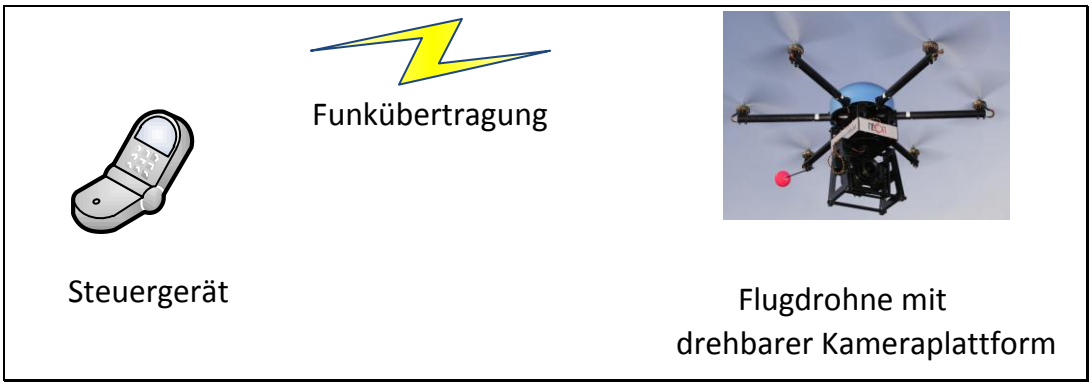


Abbildung 3-11: Übersicht Sollzustand

4 Planung und Entwicklung der Hardware

In diesem Kapitel wird die Hardwareentwicklung erläutert.

4.1 Hardwareentwicklung

Die Hardwareentwicklung begann mit der Auswahl der mechanischen Bauteile. Allem voran war es wichtig, ein passendes Gehäuse zu finden.

Bei der Gehäuseauswahl galt zu beachten:

- Die Ausführung muss robust sein, für einen Einsatz im Freien.
- Die Abmessungen sollten möglichst gering sein, aber trotzdem müssen die gesamte Elektronik mit Spannungsversorgung, die Bedienelemente und das LC-Display untergebracht werden.
- Stoßunempfindlichkeit für Feldeinsatz soll gegeben sein.

Bei einer Internetrecherche wurde nach einem Gehäuse gesucht, welches die Kriterien der angeführten Überlegungen bestmöglich erfüllt. Die Wahl fiel auf das im Anhang A abgebildete Gehäuse.
Die Auswahl des Gehäuses war ein entscheidender Schritt für den weiteren Konstruktionsverlauf. Die Abmessungen der Platine und die Bauteilanordnung sind somit vorgegeben.

Zur Dateneingabe am Kontroller sollte eine geeignete Tastatur gefunden werden. Nach ausführlichen Überlegungen fiel aus folgenden Gründen die Entscheidung eine Folientastatur zu verwenden:

- geringe Abmessungen,
- Schmutzunempfindlichkeit für Einsatz im Freien,
- am Gehäuse anliegend und somit vor Beschädigung geschützt,
- preisgünstig,
- robust;

Die Abbildung der gewählten Tastatur ist im Anhang A zu finden.

Auf die Auswahl der übrigen elektromechanischen Komponenten wird an dieser Stelle nicht genauer eingegangen. Die Datenblätter sind unter der im Anhang angeführten Literaturquelle abrufbar. Dies gilt auch für deren Abbildungen.

In der folgenden Tabelle werden jene Bauteile aufgelistet, die maßgebend für die Konstruktion der Print- und der Frontplatte relevant sind:

Bezeichnung
LCD Rahmen 2x16
Textdisplay STN grau 2x16
Schieberegler linear 470 Ohm
Folientastatur Matrix 3x4
Emmerich Akkupack 6V / 800mA

Tabelle 4-1: Bauteile bestellt bei Conrad Elektronik

Bezeichnung
Schiebeschalter 1 Wechsler 90° gew.
Gehäuse Serie 33

Tabelle 4-2: Bauteile bestellt bei RS Components

Auswahl des Mikrocontrollers:

Der Mikrocontroller muss folgende Mindestanforderungen erfüllen:

- 2 x Analogeingänge (Schieberegler)
- 1 x Serielle Schnittstelle
- 16 x I/O Leitungen
- 1 x 16 Bit Timer
- mindestens 8 kByte Programmspeicher (Flash)
- mindestens 50 Byte EEPROM - Daten

Um diesen Anforderungen gerecht zu werden, würde der Mikrocontroller ATMega16 der AVR-Familie (Firma ATMEL) entsprechen.
Als Controller wurde auf Wunsch der Fa. NeCon der ATmega32 verwendet, da dieser für mögliche Erweiterungen die nötigen Reserven bietet und der Standardcontroller in der Firma ist.

4.1.1 Schaltungsentwurf

Für den Schaltungsentwurf wurde das Softwaretool EAGLE Version 4.15 mit Schaltplan-, Layouteditor und Autorouter verwendet.

Für einen Teil der verwendeten Bauteile wurde die Bibliothek von EAGLE erweitert. (vgl. /5/ S. 209 ff.)

Beim Entwickeln des Schaltplanes galt es folgende Komponenten miteinander zu vernetzen:

- Mikrocontroller ATmega32 (DIL Version),
- LC – Display 2x16 Zeichen (mit Hintergrundbeleuchtung),
- Spannungsversorgung (5V für Controller und LCD; 3,3V für Wi.232-Modul),
- Folientastatur 3x4,
- Schieberegler (Einstellung Vertikal- und Horizontalposition),
- Wi.232-EUR Modul (für serielle Übertragung über Funk) & Programmier-schnittstelle,
- Programmierschnittstelle für ATmega32,
- RS232 Schnittstelle schaltbar (für Test, Programmentwicklung und Debug-ging);

Die einzelnen Schaltungsteile und Komponenten werden im folgenden Abschnitt genauer erläutert.

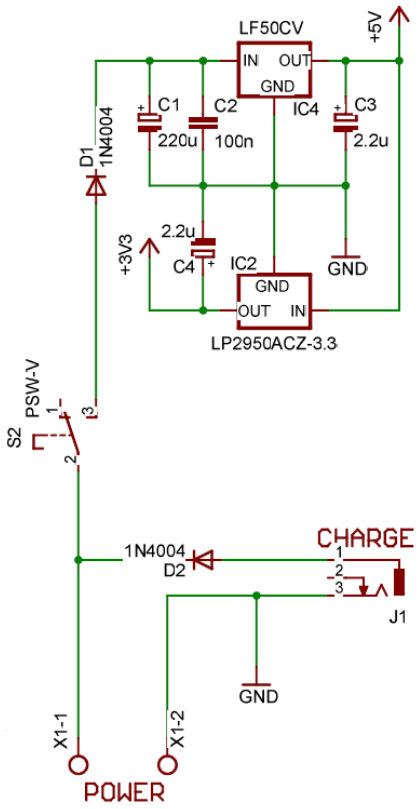
Die Spannungsversorgung

Zur Energieversorgung des Steuergerätes dient der in Tabelle 4-1angeführte Akkupack (6 Volt, 800 mA). Dieser soll mittels Ladegerät geladen werden, ohne das Steuergerät öffnen zu müssen. Aus diesem Grund wurde eine Ladebuchse eingeplant (Charge Anschluss). Zur Versorgung der Elektronik sind zwei unterschiedliche Spannungen nötig:

- 5V für Controller und LCD
- 3,3V für Wi.232-Modul

Es wurden also 2 Längsregler (LF50CV und LP2950ACZ) ausgewählt und wie in den Datenblättern beschrieben, mit Kondensatoren beschaltet.

Die Abbildung 4-12 zeigt den Schaltungsausschnitt der Spannungsversorgung des Steuergerätes:



CHARGE – Anschluss: (J1)
Mittels Akkuladegerät können die eingebauten Akkus geladen werden.

POWER – Anschluss: (X1-1, X1-2)
Über diese Anschlüsse wird das Gerät mittels Akku versorgt.

LF50CV: (IC4)
Längsregler +5V . (Siehe Datenblatt Quelle /6/)

LP2950ACZ-3.3: (IC2)
Längsregler +3,3V . (Siehe Datenblatt Quelle /6/)

Abbildung 4-12: Spannungsversorgung
(Eigenentwurf)

Über den Schalter S2 wird das Gerät eingeschaltet. Die Dioden D1 und D2 dienen als Schutz vor falscher Polarität der Spannungsversorgung.

Beschaltung des Mikrocontrollers:

Im Kapitel 2 wurde der verwendete Mikrocontroller bereits beschrieben. Nun sollen die im Projekt verwendeten Komponenten und die Beschaltung des Controllers genauer betrachtet werden.

Port I/O:
Jeder Port oder Pin kann unterschiedlich genutzt werden. Die Ports können als digitaler Ein- oder Ausgang verwendet werden. Die Steuerung erfolgt über das Data Direktion Register (DDR). Es werden immer 8 Pins zu einem Port zusammengefasst. Der ATmega 32 hat 4 Ports (PA, PB, PC, PD) zu je 8 Pins und somit 32 Pins.

Alle Ports/Pins werden über Register gesteuert. Jedem Port sind 3 Register zugeordnet.

- Datenrichtungsregister DDRx
- Ausgangsregister PORTx
- Eingangsadressen PINx

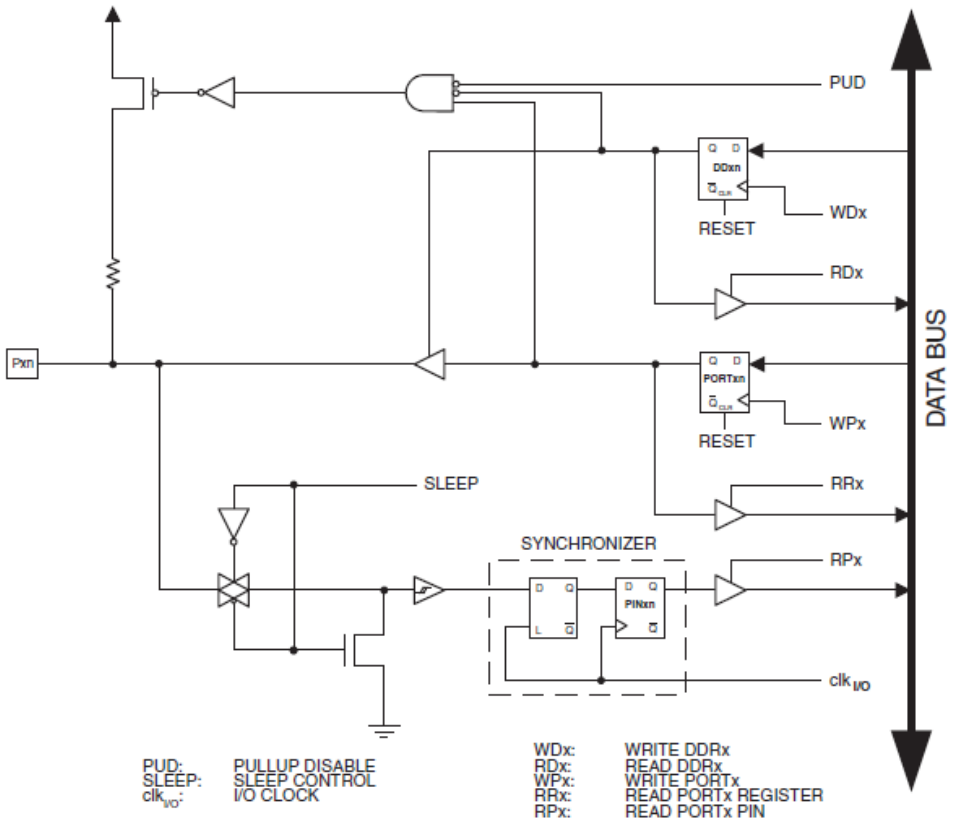


Abbildung 4-13: I/O – Ports des ATmega32
(Quelle:/7/)

Die Port-Pins sind multifunktionell und können je nach Bedarf genutzt werden (Siehe Abbildung 4-14).

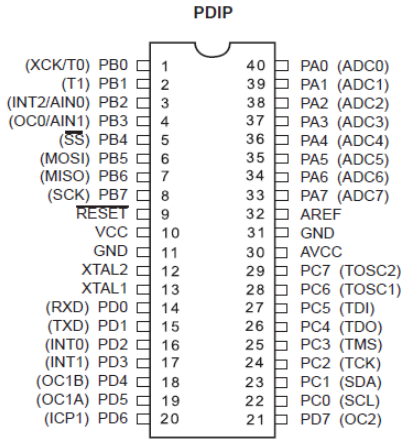


Abbildung 4-14: Gehäuse mit Pin-Belegung – Mikrocontroller ATmega32 (Quelle /7/)

Analog/Digital-Converter:

Die manuelle Steuerung der Kameraplattform erfolgt mit Schiebereglern in Form einer Analogspannung. Es werden also zwei Analogeingänge des Mikrocontrollers dafür benötigt (ADC0 und ADC1).

Die Eigenschaften der Analogeingänge des ATMega 32 kurz zusammengefasst:

- 10 Bit Auflösung
- Wandlungsverfahren: sukzessive Aproximation
- Wandlungszeit: 13 – 260 µs
- 8 Analogeingänge (multiplex)
- Eingangsspannung 0 – Vcc

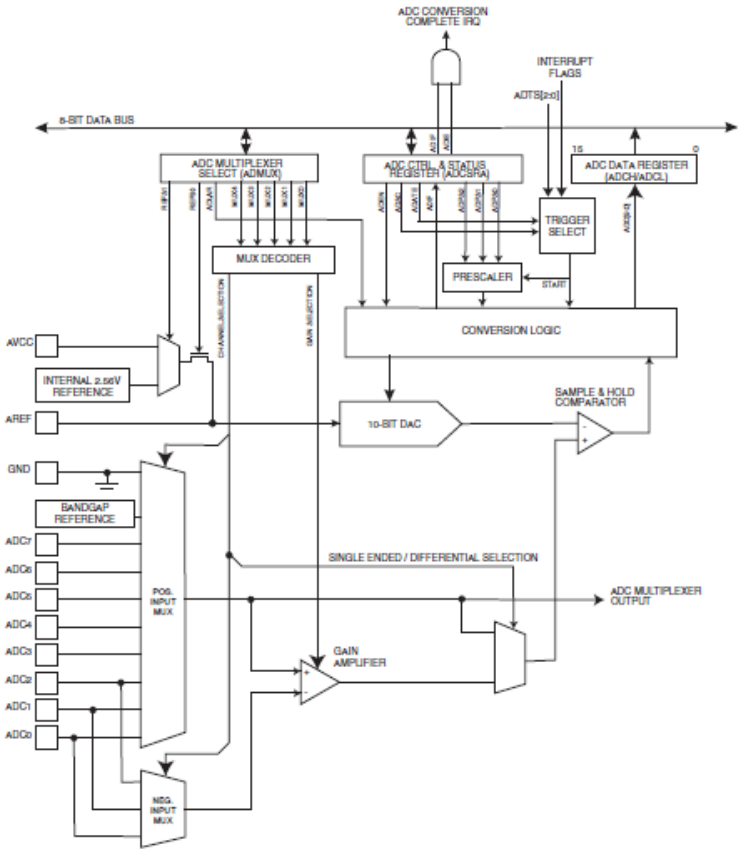


Abbildung 4-15: Blockschaltung Analog/Digital - Converter (Quelle:/7/)

Die folgende Abbildung zeigt einen Schaltungsausschnitt aus dem Schaltungsentwurf mit der Minimalbeschaltung des Mikrocontrollersystems, welche im Wesentlichen aus

- Reset- Beschaltung
- Takterzeugung (externer Quarz 10 MHz)
- und Spannungsversorgung

besteht.

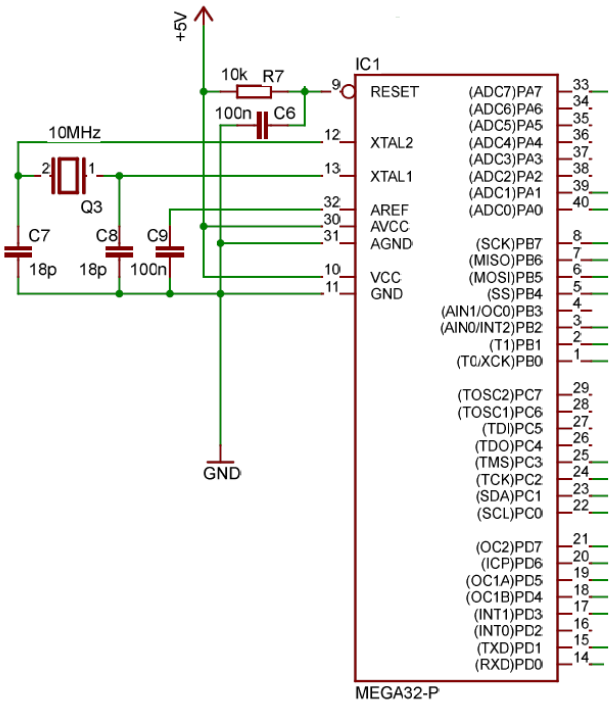


Abbildung 4-16: Beschaltung – Mikrocontroller ATmega32 (Eigenentwurf)

UART:

Wie die meisten AVR Controller besitzt auch der ATMega32 einen UART (Universal Asynchronous Receiver and Transmitter). Das ist eine serielle Schnittstelle, die zur Datenübertragung zwischen Mikrocontroller und PC, bzw. anderen Peripheriekomponenten genutzt werden kann. Zur Übertragung werden zwei Pins am Controller benötigt: TXD und RXD (Pin 14 und Pin 15, vgl. Abb.4-14). Über TXD ("Transmit Data") werden Daten gesendet und über RXD ("Receive Data") empfangen.
Bei der Diplomarbeit wird am UART das Funkmodul Wi.232 für den seriellen Datenaustausch mit dem Receiver angeschlossen.

Das Funkmodul Wi.232 wurde im Abschnitt 2 bereits genauer beschrieben.

Die folgende Abbildung zeigt wie das Funkmodul mit dem Mikrocontroller im Schaltungsentwurf verbunden ist. Mit dem Schalter S1 können die Leitungen RXD und TXD des Funkmodules zu- und weggeschaltet werden. Dann ist es möglich das Sendegerät direkt mittels RS232 Schnittstelle mit dem Empfangsmodul zu verbinden und somit kann für Testzwecke auf die Funkübertragung verzichtet werden.

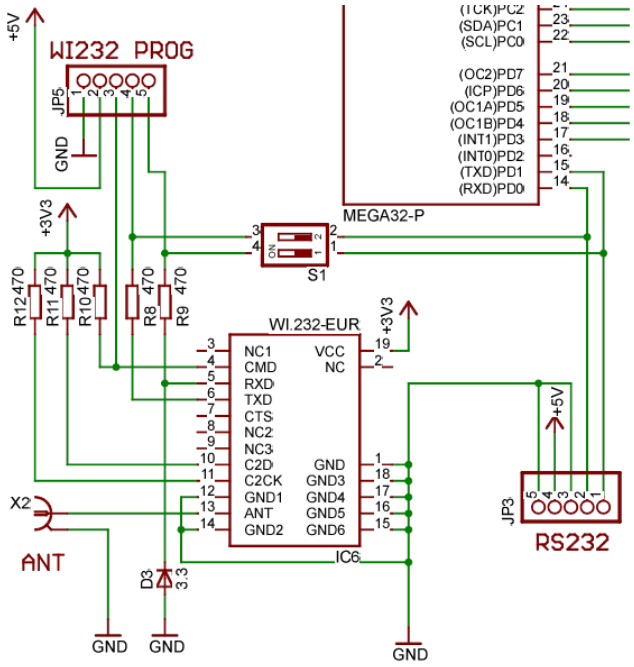


Abbildung 4-17: Beschaltung des WI.232
(Eigenentwurf)

Anschluss des Displays:

Die Anschlussbelegung des LC-Displays wurde im Abschnitt 2 bereits beschrieben. Die Umsetzung im Projekt ist in Abbildung 4-18 ersichtlich. Das LCD wird im 4 – Bit – Modus betrieben. Es sind 4 Datenleitungen notwendig (DB4 – DB7). Diese sind mit dem Controller über Port D, Pin 4 bis Pin 7 zusammengeschaltet. Die Enable- und der Register-Select-Anschlüsse wurden auch mit dem Mikrocontroller verbunden (PinC.1 und PinC.0). Der R/W-Anschluss (read/write) des LC-Displays muss, bei der Ansteuerung mit BASCOM, auf Masse (GND) gelegt werden.

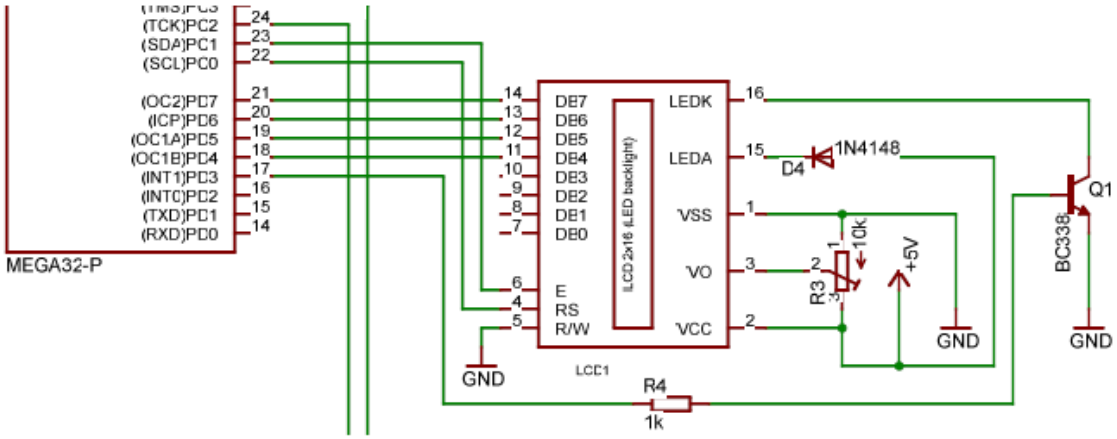


Abbildung 4-18: Schaltungsausschnitt mit LC-Display

Die Hintergrundbeleuchtung kann über den Transistor Q1 (Transistor als Schalter) mittels Mikrocontroller Port D /Pin 3 geschaltet werden. Der Widerstand R4 (1K) dient der Strombegrenzung des Basisstromes von Transistor Q1.

Folientastatur 3x4

Die Folientastatur wird wie in Abb. 4-19 dargestellt mit dem Controller verbunden. Die Tastatur hat 12 Tasten. Um Leitungen zu sparen, sind die Tasten in einer Matrix angeordnet (3 Spalten und 4 Zeilen). Zum Schutz der Portleitungen des Mikrocontrollers wird ein Widerstandsnetzwerk in Reihe geschaltet. Die Auswertung, welche der Tasten gedrückt wird, kann mit dem Mikrocontroller durchgeführt werden (vgl. /1/ S. 205 ff.) und wird im Kapitel Softwareentwicklung näher erläutert.

Schieberegler (Vertikal und Horizontal):

Die Schieberegler R1 und R2 werden mit den Analogeingängen (ADC0 und ADC1) des Mikrocontrollers verbunden (siehe Abb. 4-19). Die Auswertung der Position erfolgt softwaremäßig durch den Mikrocontroller. (vgl. /1/ S.284 ff.). Die den Schie-

bereglern vorgeschalteten Trimmpotentiometer R13 und R14 dienen als Spannungsteiler zur Voreinstellung der Eingangsspannung an den Analogeingängen.

ISP (In-System-Programmer):

Der ISP-Anschluss dient als Schnittstelle zur Programmierung und Debuggen. Damit ist es möglich den Mikrocontroller zu programmieren oder Softwareänderungen durchzuführen ohne diesen aus dem Steuergerät entfernen zu müssen.

Beim verwendeten Programmer AVR ISP mkII ist ein Hardware-Debuggen leider nicht möglich. Aus diesem Grund wurden verschiedene Tests überlegt, welche im Kapitel 6 näher ausgeführt werden.

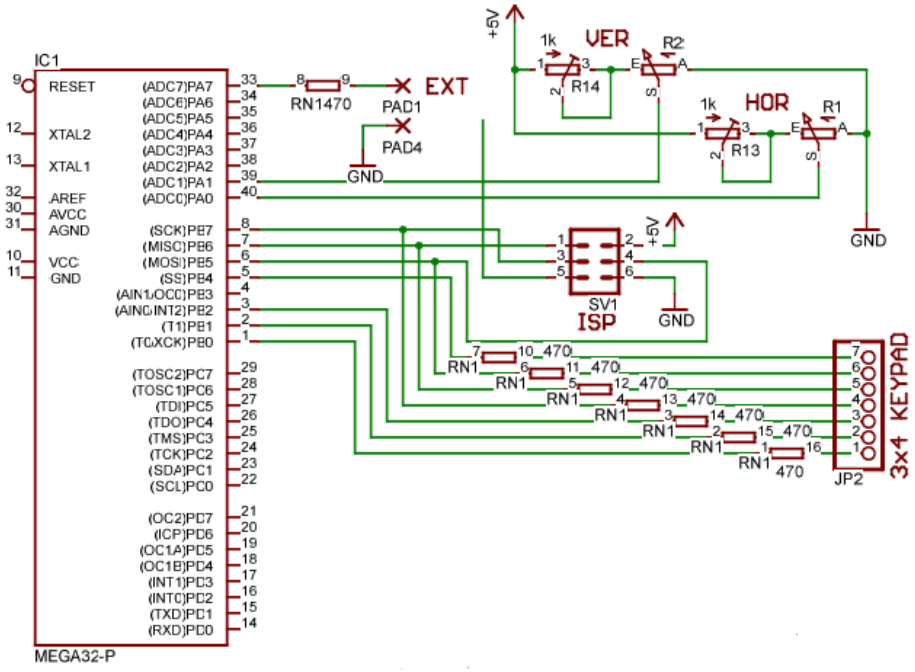


Abbildung 4-19: Schaltungsausschnitt mit Eingabe-Beschaltung des µC

EXT-Anschluss:

Dieser Anschluss hat keine Funktion und ist lediglich als Reserve für eventuelle Erweiterungen vorgesehen.

LED – Ausgabe:

Als zusätzliche Ausgabe werden zwei Leuchtdioden LED1 (Status-LED) und LED2 (Busy-LED) verwendet. Die Ausgänge des Mikrocontrollers können bis zu 20mA

aufnehmen, dadurch ist es möglich die LEDs direkt anzusteuern. Diese sind am Port C (Pin 2 und Pin 3) des Controllers angeschlossen (siehe Abb. 4-20).

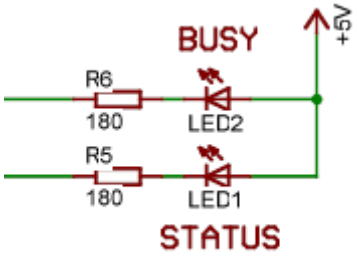


Abbildung 4-20: LED - Ansteuerung

Der Strom durch die Leuchtdioden wird mit den Vorwiderständen R5 und R6 (180Ω) begrenzt. Um die Beschaltung der Pins des Mikrocontrollers richtig dimensionieren zu können ist es notwendig, die Beschaltung der I/O-Ports etwas näher zu betrachten (Abb. 4-13). Wenn der Ausgang des Mikrocontrollers aktiv ist, wird der n-Kanal-FET gegen Masse leitend und es kann ein Strom (der gegen +5 Volt geschalteten Dioden) gegen Masse abfließen.

Der Vorwiderstand der Dioden ist so zu dimensionieren, dass der maximale Strom von 20mA nicht überschritten wird.

Der Strom durch die LEDs beträgt:

$$I_{LED} \sim \frac{5V - U_F - U_{DS}}{R_V} \sim \frac{5V - 2V - 0{,}8V}{180\Omega} \sim 12mA$$

U_F = Spannungsabfall an der LED (2V bei LED rot und 2,2V bei LED grün)

U_{DS} = Spannungsabfall am FET (ca. 0,8V)

Der Strom von ca. 12mA reicht um die LED zu betreiben.

Wenn man größere Lasten schalten will, ist es notwendig einen externen Schalttransistor zu verwenden, wie bei der Ansteuerung für die Hintergrundbeleuchtung des LD-Displays (siehe Abb. 4-18).

Gesamtschaltung:

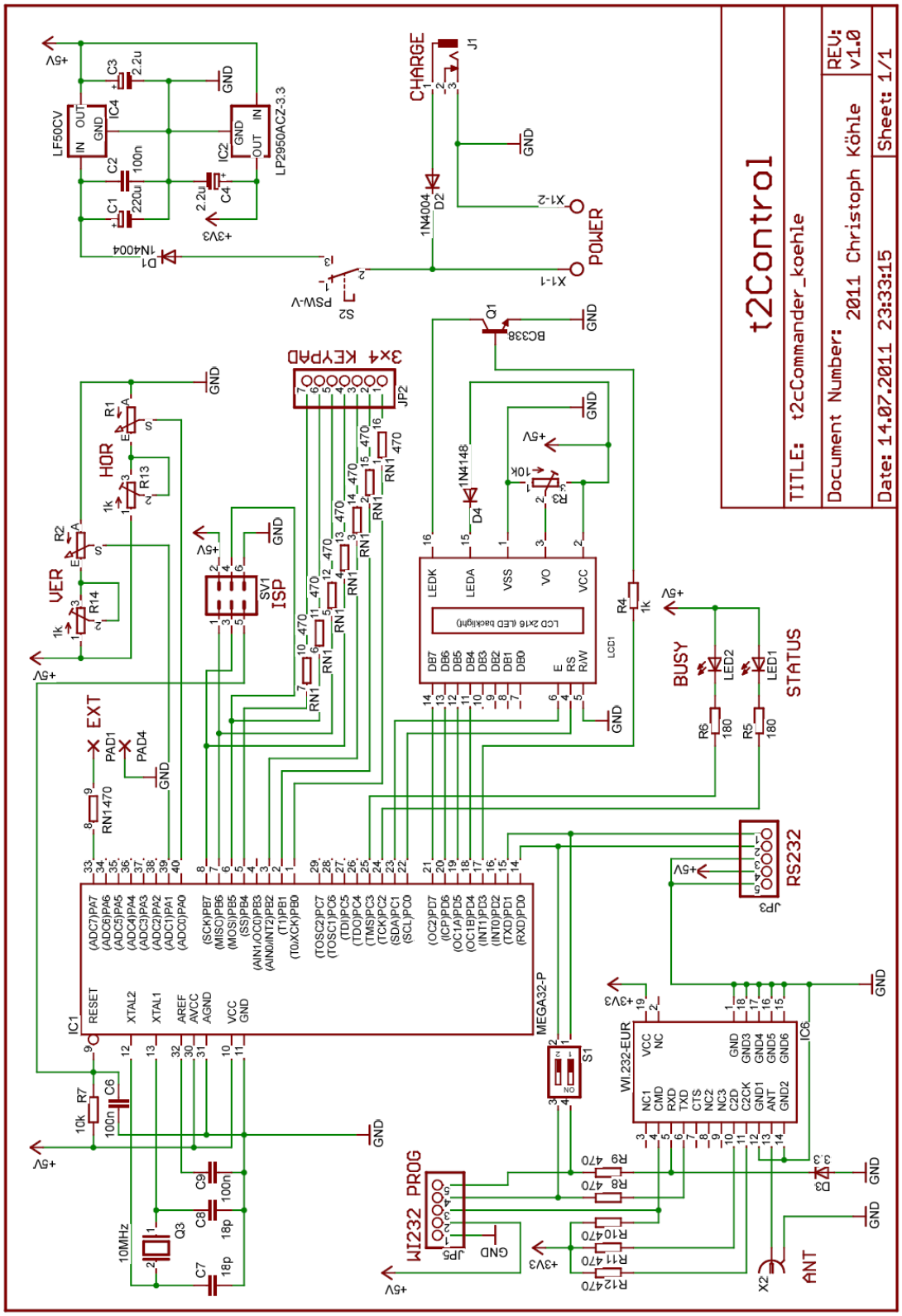


Abbildung 4-21: Schaltplan erstellt mit EAGLE

4.1.2 Platinen - Layout

Bei der Konstruktion der Leiterplatte wurde zuerst die Leiterplattengröße aufgrund der Gehäuseabmessungen (Frontplattenausschnitt) festgelegt. Dann wurde die Platzierung jener Bauteile vorgenommen, die über die Frontplatte zu bedienen bzw. durchzuführen sind (Schieberegler, Leuchtdioden), wobei gleichzeitig auch die Frontplatte konstruiert werden musste (siehe Abbildung 4-22).

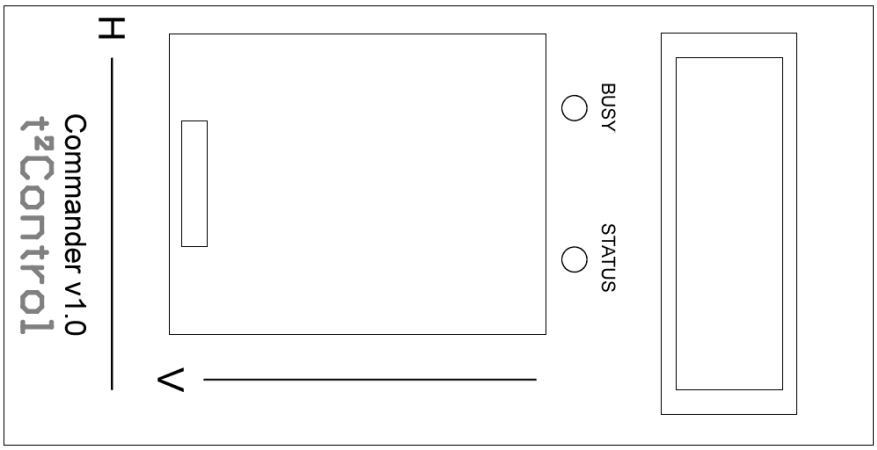


Abbildung 4-22: Frontplatte des Steuergerätes

Die Ladebuchse und der Ein- Ausschalter wurden stirnseitig auf der Unterseite angebracht. Entscheidend für die Qualität des späteren Layouts ist die Platzierung der übrigen Bauteile auf der Platine. Je nach Erfahrung bedarf dieser Vorgang einiger Versuche um die geeignete Anordnung zu finden. Nach der Platzierung der restlichen Bauteile konnte die Platine mittels Autorouter geroutet werden.

Das Ergebnis des Autorouters wurde dann noch manuell optimiert.

Im Weiteren folgen Bilder der Platine bzw. Fertigungsunterlagen.

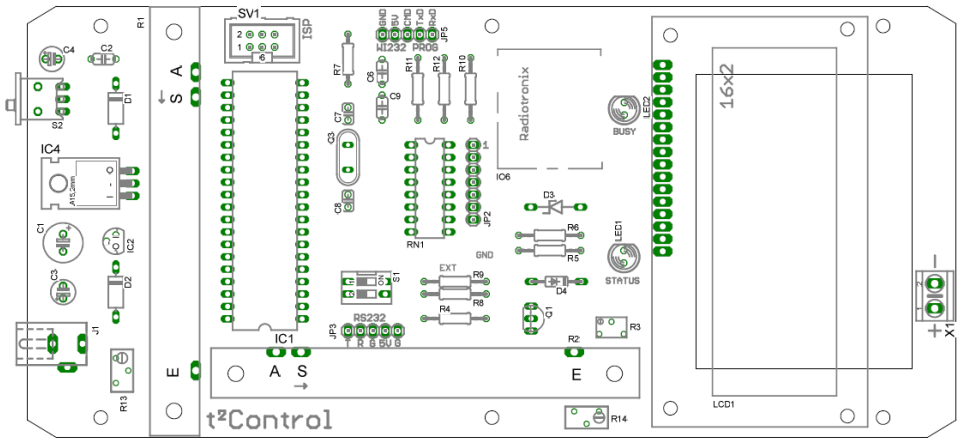


Abbildung 4-23: Print mit Bauteilen

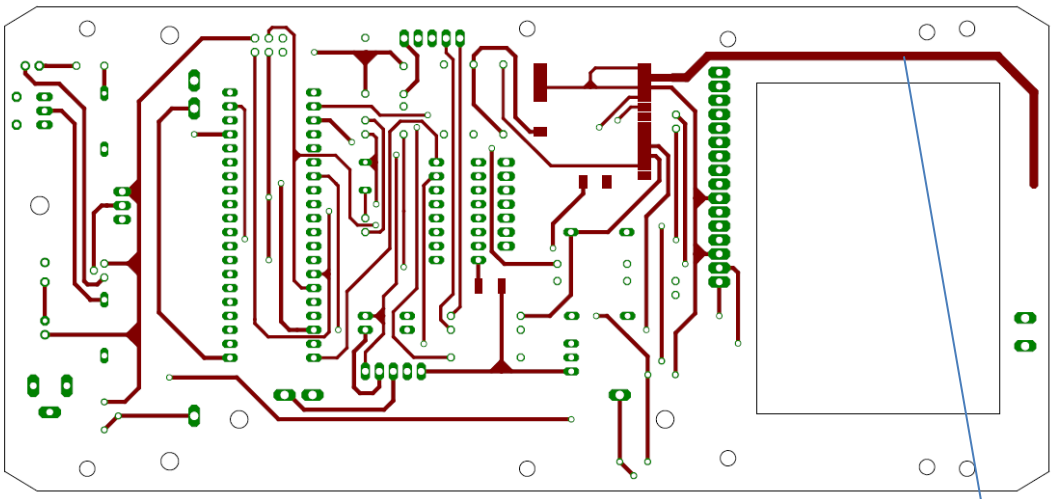


Abbildung 4-24: Print mit Leiterbahnen Top – Layer (Oberseite)

Die Antenne wird als Kupferleiterbahn ausgeführt (siehe Abbildung 4-24). Die Länge der Leiterbahn wurde mit $\frac{\lambda}{4}$ wie folgt berechnet:

Sendefrequenz des Wi.232 : f = 868MHz

Lichtgeschwindigkeit: c = 299 792 458 m/s

Wellenlänge: $\lambda = \frac{c}{f}$ = 0,34m => Antennenlänge $\frac{\lambda}{4}$ **= 85mm**

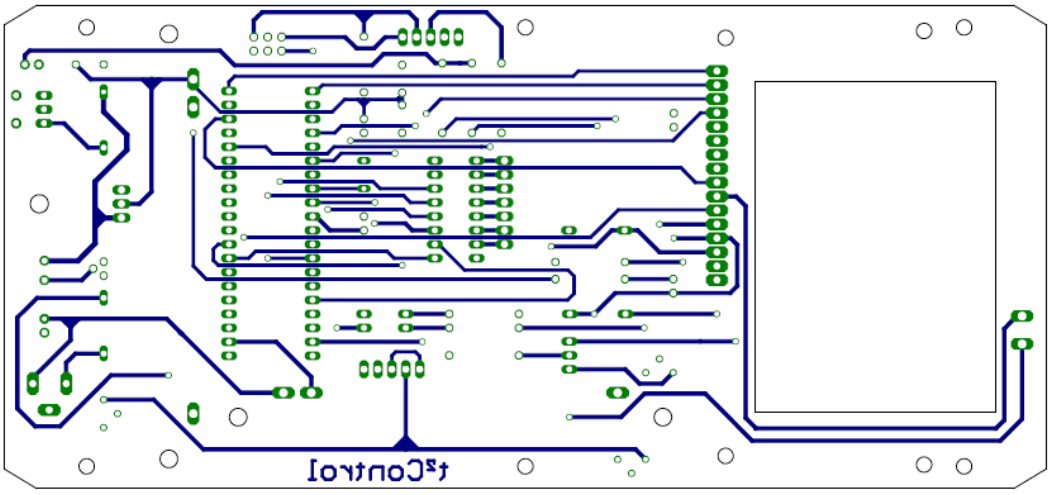


Abbildung 4-25: Print mit Leiterbahnen Bottom – Layer (Unterseite)

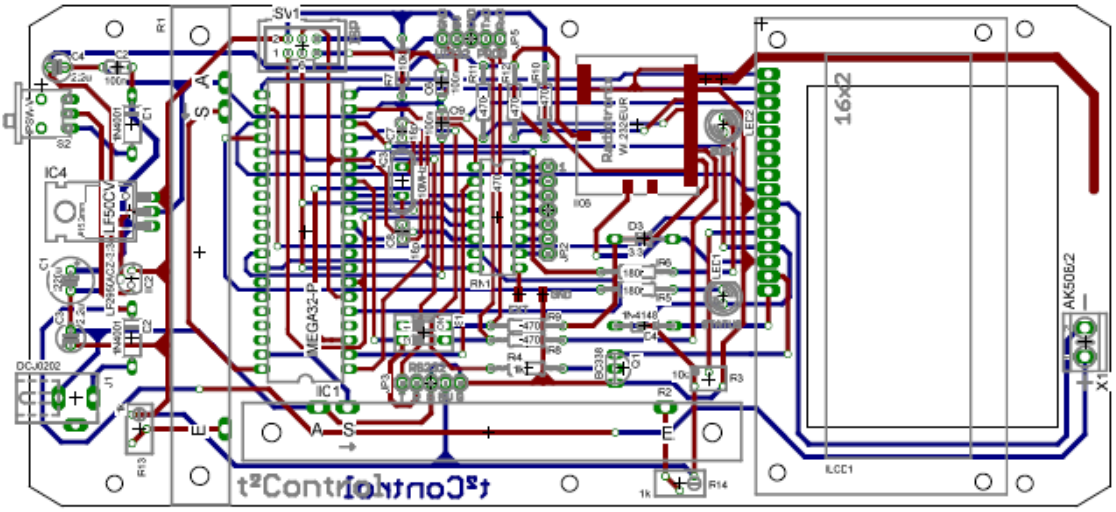


Abbildung 4-26: Print komplett

Verwendete Bauteile

Bezeichnung	Benötigt
Schiebereglerknopf schwarz	2
LED 5 mm rot	1
LED 5 mm grün	1
ATMEGA32 DIL-40	1

Bezeichnung	Benötigt
LCD Rahmen 2x16	1
Textdisplay STN grau 2x16	1
Schieberegler linear 470 Ohm	2
Folientastatur Matrix 3x4	1
Messerleiste 6polig	1
Emmerich Akkupack 6V / 800mA	1
Transistor BC338	1
Ladegerät NiMH 300mA	1
Widerstand 100 Ohm	2
Widerstand 470 Ohm	5
Widerstand 1k	1
Widerstand 10k	1
Schraube M2,5 x 20mm	4
Schiebeschalter 1 Wechsler 90° gew.	1
Gehäuse Serie 33	1
Printklemme RM 5,08 2 polig	1
IC Sockel 40 polig	1
Quarz HC49S 10 MHz	1
Trimmerpoti 1k	2
Trimmpoti 10k	1
Abstandshalter 12,7 mm	4
Widerstandsnetzwerk 470 Ohm	1
Spannungsregler LV50CV 5V / 0,5A	1
Spannungsregler LP2950ACZ -3-3	1
DIP Switsch 2 polig	1
Kondensator 22 pF	2
Kondensator 100n	3
Elko 220 uF	1
Tantal-Elko 2,2 uF	2
Z-Diode 3,3V	1
Diode 1N4004	2
Diode 1N4148	1

Tabelle 4-3: Bauteilliste gesamt

Vor dem Bestücken der Platine, wurde diese in das Gehäuse eingebaut (siehe Abbildung 4-27) um alle Maße für die Gehäuseausschnitte zu kontrollieren und eventuelle Korrekturen vorzunehmen. Die Maße für die Abstände der Leuchtdioden, Schieberegler und das LC-Display konnten so am Gerät direkt gemessen werden um die entsprechenden Abstandhalter für die Montage auszuwählen.

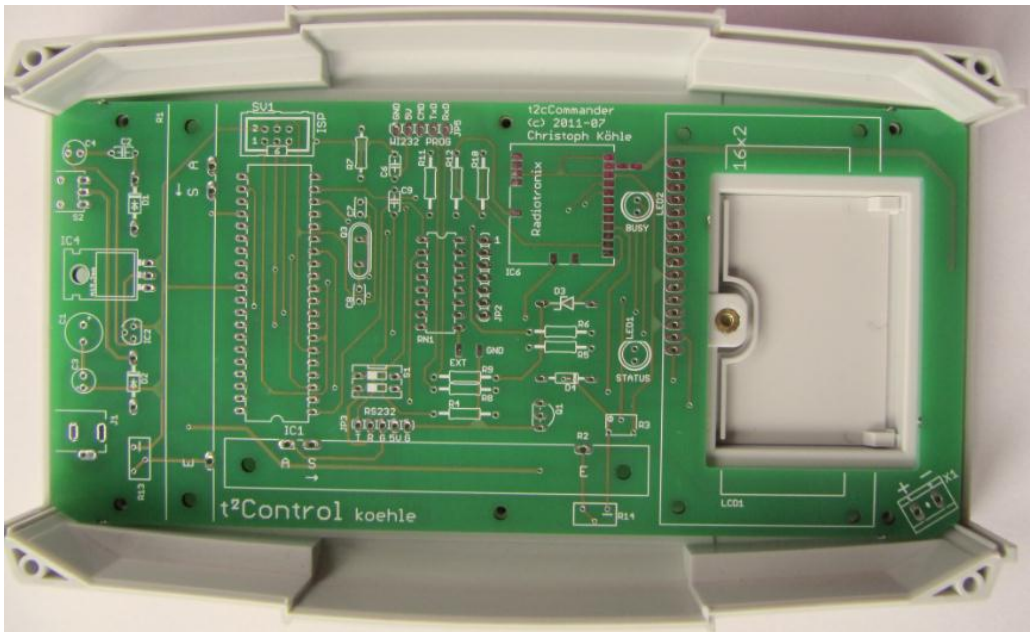


Abbildung 4-27: Printplatte ohne Bestückung

Die Abbildung 4-28 zeigt die fertig bestückte Platine im Gehäuse eingebaut und somit bereit für den Test.

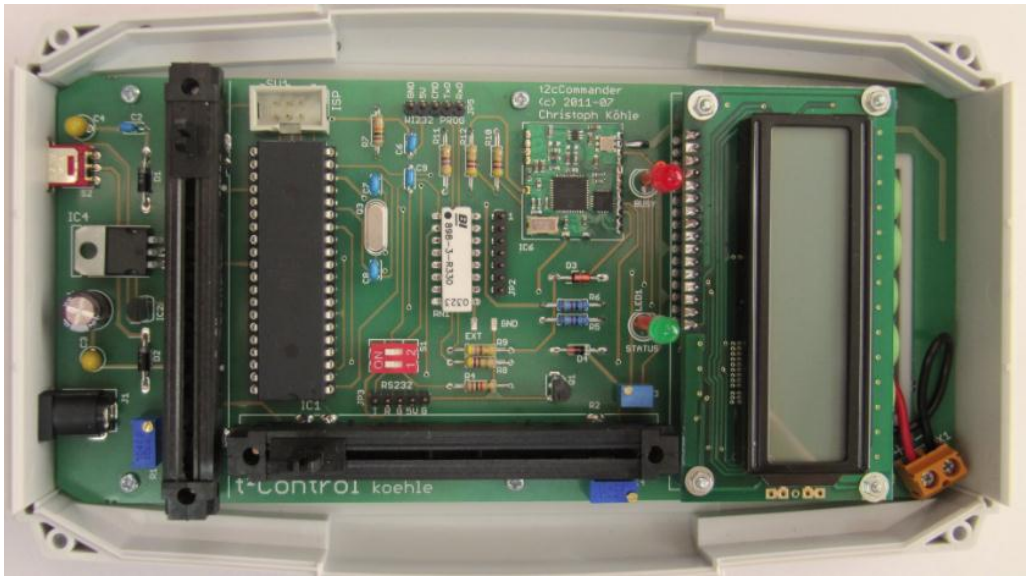


Abbildung 4-28: Printplatte mit Bestückung

Das fertige Handsteuergerät mit der Bezeichnung T^2 Control (turn table control) ist in Abbildung 4-29 zu sehen. Mit der gewählten Anordnung der Komponenten (Schieberegler, Tastatur und Display) an der Frontplatte soll das Steuergerät den gestellten Anforderungen, der einfachen Bedienbarkeit und Ergonomie gerecht werden.

Abbildung 4-29: Commander T^2Control

5 Entwicklung der Software

Zur Programmierung wird die Entwicklungsumgebung BASCOM-AVR von MCS Electronics verwendet. Auf der Website von MCS Electronics kann die Demoversion unter URL:

www.mcselec.-com/index.php?option=com_docman&task=download&gid=139&itemid=54

heruntergeladen werden. Bei der BASCOM-AVR Demo ist die Größe des generierten Codes auf 4KByte beschränkt. Um spätere Probleme zu vermeiden wurde von der Fa. NeCon dankenswerterweise die Vollversion zur Verfügung gestellt.

Da BASCOM-AVR den verwendeten Programmer AVR ISP mkII nicht unterstützt, wird zusätzlich AVR Studio 4 zur Übertragung zum Mikrocontroller verwendet.

5.1 Softwarefunktionalität

Grundlage für die Softwareerstellung ist die Liste der Befehle welche an die Kameraplattform gesendet werden sollen. Diese wurde in Absprache mit der Fa. NeCon erstellt:

Automatik-Modus horizontal 5 Grad Schrittweite
Automatik-Modus horizontal 10 Grad Schrittweite
Automatik-Modus horizontal 20 Grad Schrittweite
Automatik-Modus horizontal 30 Grad Schrittweite
Automatik-Modus horizontal 40 Grad Schrittweite
Automatik-Modus horizontal 72 Grad Schrittweite
Abfrage Ablaufprogramme
Horizontale Position manuell vorgeben [000 .. 720]
000 -> Null-Position (home): 0 GRD
720 -> Maximal-Position: 360 GRD
Vertikale Position manuell vorgeben [000 .. 720]

Tabelle 5-4: Kommandos an Receiver

Die Befehle aus Tabelle 5-4 werden mit der am Handsender angebrachten Folientastatur eingegeben und an das Empfängermodul mit dem Wi.232 – Modul über Funk gesendet. Das Wi.232 wird vom Mikrocontroller wie die serielle Schnittstelle (RS232) behandelt und es werden die entsprechenden BASCOM – Befehle verwendet.

Auswertung der Folientastatur:

Bei der Tastatur handelt es sich um eine Matrix-Tastatur. Die Auswertung ist mit dem BASCOM – Befehl „*getkbd*“ möglich. Wie in der BASCOM- Hilfe und Literaturquelle /13/ (Seite 104 ff.) beschrieben, ist mit dem Befehl „*getkbd*“ das Auslesen einer 4x4 Tastaturmatrix möglich (siehe Abbildung 5-30). Da nur eine 3x4 Matrixtastatur verwendet wird, bleibt der Portpin PB.3 am ATMega 32 frei (siehe Abbildung 4-19).

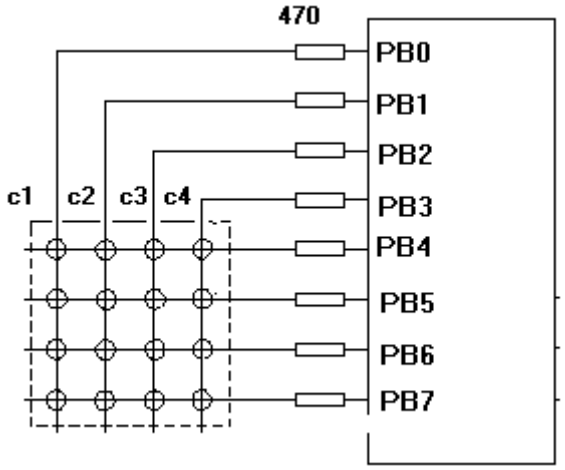


Abbildung 5-30: Anschluss der Tastatur an den Mikrocontroller
(Quelle: BASCOM-Hilfe)

Die Tastaturbefehle wurden wie folgt festgelegt:

Taste 1: Automatik-Modus horizontal 5 Grad Schrittweite
Taste 2: Automatik-Modus horizontal 10 Grad Schrittweite
Taste 3: Automatik-Modus horizontal 20 Grad Schrittweite
Taste 4: Automatik-Modus horizontal 30 Grad Schrittweite
Taste 5: Automatik-Modus horizontal 40 Grad Schrittweite
Taste 6: Automatik-Modus horizontal 72 Grad Schrittweite
Taste 7: Reserve
Taste 8: Reserve
Taste 9: Abfrage Ablaufdiagramme
- 9 -> 1 Abfrage Ablaufdiagramm #1
- 9 -> 2 Abfrage Ablaufdiagramm #2
- 9 -> 3 Abfrage Ablaufdiagramm #3
- 9 -> 4 Abfrage Ablaufdiagramm #4
- 9 -> 5 Abfrage Ablaufdiagramm #5
- 9 -> 6 Abfrage Ablaufdiagramm #6

Taste 0: Relais aktivieren (Kamera auslösen)
Taste *: Abbruch (Ausstieg aus den Untermenüs)
Taste #: LCD-Hintergrundbeleuchtung EIN / AUS

Bemerkung: Bei der Taste 9 handelt es sich um einen „Zwei – Tastenbefehl“. Das heißt, nach dem Drücken der Taste 9, muss ein weiterer Tastenbefehl (1…6) folgen um das Ablaufprogramm auszuwählen.

Auswertung der Schieberegler:

Die Schieberegler für horizontales bzw. vertikales Verstellen der Kameraplattform erfolgt mit den Analog –Digital - Convertern ADC0 (Horizontal) und ADC1 (Vertikal) mittels BASCOM – Befehl „*getadc(Kanal)*“. Mit „*getadc()*“ wird ein Analogwert eines analogen Eingangspins eingelesen. Da es sich um 10-Bit AD-Converter handelt, ist das Ergebnis zwischen 0 und 1023. Der Variablentyp zur Aufnahme des Wertes kann Word oder Integer sein, eine Byte-Variable reicht nicht aus.

Wie bereits im Abschnitt Hardwareentwicklung beschrieben wurde den Schiebereglern jeweils ein Trimmpotentiometer als Spannungsteiler vorgeschaltet, sodass die Eingangsrohwerte einen Maximalwert von 720 erreichen können (siehe Abb.4-23). Das heißt der Maximalwert von 720 entspricht einem Winkel von 360° und der Minimalwert 0 entspricht somit dem Winkel 0°. Die Auflösung für einen Schritt entspricht einem Winkel von 0,5°.

Beispiel aus Quelltext:

```
Config Adc = Single , Prescaler = 16 , Reference = Internal
Start Adc                       'ADC einschalten
Adw = Getadc(0)                 'ADC am Beginn 2x einlesen
Adw = Getadc(0)                 'horizontaler Analogwert
Cls
If Adw > 0 Then
   Row1 = "Schieberegler in"
   Row2 = "Nullpos. bringen"
   Display_text
Else
……
```

Text - Ausgabe an LC – Display:

Das Display kann in BASCOM mit wenigen Befehlen angesprochen werden. Mit „*Config Lcdpin = Pin,…*“ werden die Anschlüsse definiert. Mit dem Befehl „*Config Lcd = Zeichen*Zeilen“* wird die Art des Displays festgelegt und mit *Lcd* „Text“ erfolgt die Ausgabe.

Beispiel aus Quelltext:

```
Config Lcdpin = Pin , Db4 = Portd.4 , Db5 = Portd.5 , Db6 =
Portd.6 , Db7 = Portd.7

Config Lcdpin = Pin , Rs = Portc.0 , E = Portc.1

Config Lcd = 16 * 2              '16 Char, 2 Zeilen

ClsCursor Off Noblink            'Cursor aus und nicht blinken
```

Binäre Ausgabe an LED:

Die Pins an denen die LED’s angeschlossen sind, müssen als Ausgang (Output) definiert werden. Der Befehl dafür lautet *„Config Pin x.y = Output“* (oder Input für Eingang). Bei den Ausgangsvariablen werden ALIAS – Namen verwendet, um den Quellcode leichter lesbar zu machen.

Beispiel aus Quelltext:

```
Config Pinc.2 = Output

Statusled Alias Portc.2

Statusled = 1                    ' Status-LED ausschalten

Config Pinc.3 = Output

Busyled Alias Portc.3

Busyled = 1                      ' Busy-LED ausschalten

Config Pind.3 = Output

Lcdbacklight Alias Portd.3

Lcdbacklight = 1     ' LCD - Hintergrundbeleuchtung ausschalten
```

5.1.1 Flussdiagramme

Ausgehend von der Funktionalität die vom Steuergerät (Commander) erfüllt werden muss, wurde die Funktion in kleinere Teilaufgaben zerlegt und diese dann einzeln gelöst um eine übersichtliche Programmstruktur zu erhalten.

Die folgenden Flussdiagramme beschreiben den globalen Ablauf der Software-Funktionen im Zusammenhang mit dem User-Interface des Commanders.

Details zu den Unterprogrammen werden als Kommentare im Quelltext kommentiert.

Initialisierung mit Verbindungsaufbau:

Nach dem Einschalten des Steuergerätes müssen alle Peripherie–Komponenten initialisiert werden (I/O Pins, Tastatur, LC-Display, Analog-Digital-Converter, …). Am Display wird eine Startmeldung ausgegeben, um die Betriebsbereitschaft zu signalisieren. Anschließend versucht der Commander sich mit dem Receiver zu verbinden. Am Display wird die Meldung „mit Receiver verbinden“ ausgegeben. Nach erfolgreichem Verbindungsaufbau, wird mit der Prüfung der Slider-Positionen fortgefahren. Gelingt der Verbindungsaufbau nicht, wenn zum Beispiel der Empfänger nicht eingeschaltet, oder die Funkverbindung unterbrochen ist, soll die Meldung „ Kein Receiver gefunden“ ausgegeben und das Programm beendet werden.

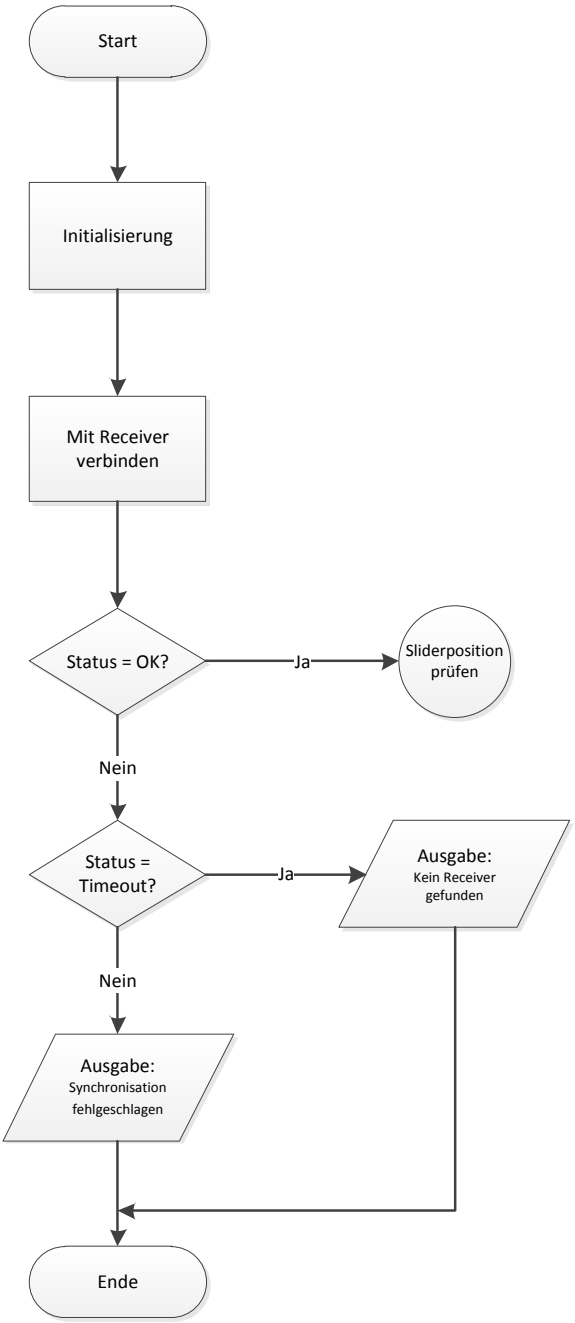


Abbildung 5-31: Flussdiagramm: Initialisierung / Verbindungsaufbau

Slider-Positionen prüfen:

Um zu verhindern, dass die Kameraplattform nach dem Verbindungsaufbau ungewollte Bewegungen ausführt, muss sichergestellt werden, dass die Schieberegler (Slider) in Null-Position sind. Dies wird durch das Einlesen der entsprechenden Analogeingänge realisiert. Zuerst wird der Wert des Horizontalreglers eingelesen. Wenn der Wert nicht Null ist, wird der Benutzer über eine entsprechende Displayausgabe aufgefordert, den Schieberegler in Nullposition zu bringen. Der Vorgang wird solange wiederholt, bis die Null-Position eingestellt ist. Der gleiche Vorgang wird nun für den Vertikal-Schieberegler durchgeführt. Erst wenn beide Schieberegler in Null-Stellung sind, wird im Programmablauf fortgefahren und der Sender ist betriebsbereit.

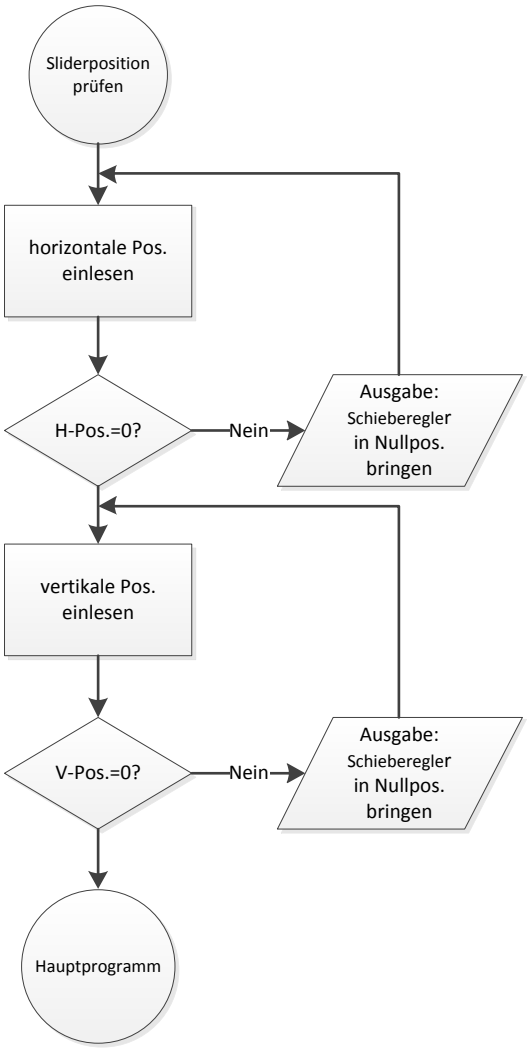


Abbildung 5-32: Flussdiagramm Slider-Position prüfen

Hauptprogramm:

Der Benutzer hat die Möglichkeit die Kameraposition manuell (über Schieberegler) oder über ein durch die Tastatur ausgewähltes Ablaufprogramm automatisch zu steuern. Der Schieberegler für die horizontale Positionsvorgabe wird eingelesen. Nachdem das Bewegen des Schiebereglers im Vergleich zum Programmdurchlaufzyklus ein sehr langsamer Prozess ist, muss vor dem Senden des Positionswertes an den Receiver geprüft werden, ob der Schieberegler noch in Bewegung ist. Dies geschieht indem die Differenz zum jeweiligen Vorgängerwert gebildet wird. Wenn die Differenz zum Vorgängerwert größer 4 ist, befindet sich der Schieberegler noch in Bewegung. Ist dies nicht der Fall, wird der aktuelle Horizontalwert als gültiger Positionswert erkannt und kann, durch Übergabe an die Hauptschleife 1, gesendet werden. Wenn keine Horizontalwertvorgabe erfolgt wird der gleiche Vorgang für die Vertikalposition durchgeführt. Der Vertikalwert wird an die Hauptschleife 2 übergeben.

Sollte weder der Horizontal- noch der Vertikalregler bewegt werden, wird auf eine Tastatureingabe getestet. Bei positiver Testung erfolgt eine Tastaturauswertung, ansonsten wird wieder zur Schieberegler- Auswertung verzweigt.

Hauptschleife 1:

Der Hauptschleife 1 wird der zu sendende Horizontalwert übergeben. Dieser wird am Display ausgegeben und zum Receiver gesendet. Anschließend wird auf eine Statusmeldung des Receivers gewartet. Während die Kameraplattform beschäftigt ist, soll die „Busy-Led“ am Steuergerät leuchten. Wenn die empfangene Statusmeldung „OK“ ist, kann davon ausgegangen werden, dass der Wert ordnungsgemäß abgesetzt wurde und die Kamerasteuerung die eingestellte Position angefahren hat. Die „Busy-Led“ wird ausgeschaltet und die Funktion verlassen.

Sollte die Statusmeldung „Busy“ empfangen werden, heißt das, die Kameraplattform ist noch beschäftigt. Es wird erneut gewartet, bis eine neue Statusmeldung oder „OK“ empfangen wird. Wenn der Status „OK“ immer noch nicht erreicht ist, wird der Status am Display ausgegeben und die Kommunikation zum Receiver geprüft. Der Horizontalwert wird erneut gesendet, bis der Wert ordnungsgemäß abgesetzt werden kann.

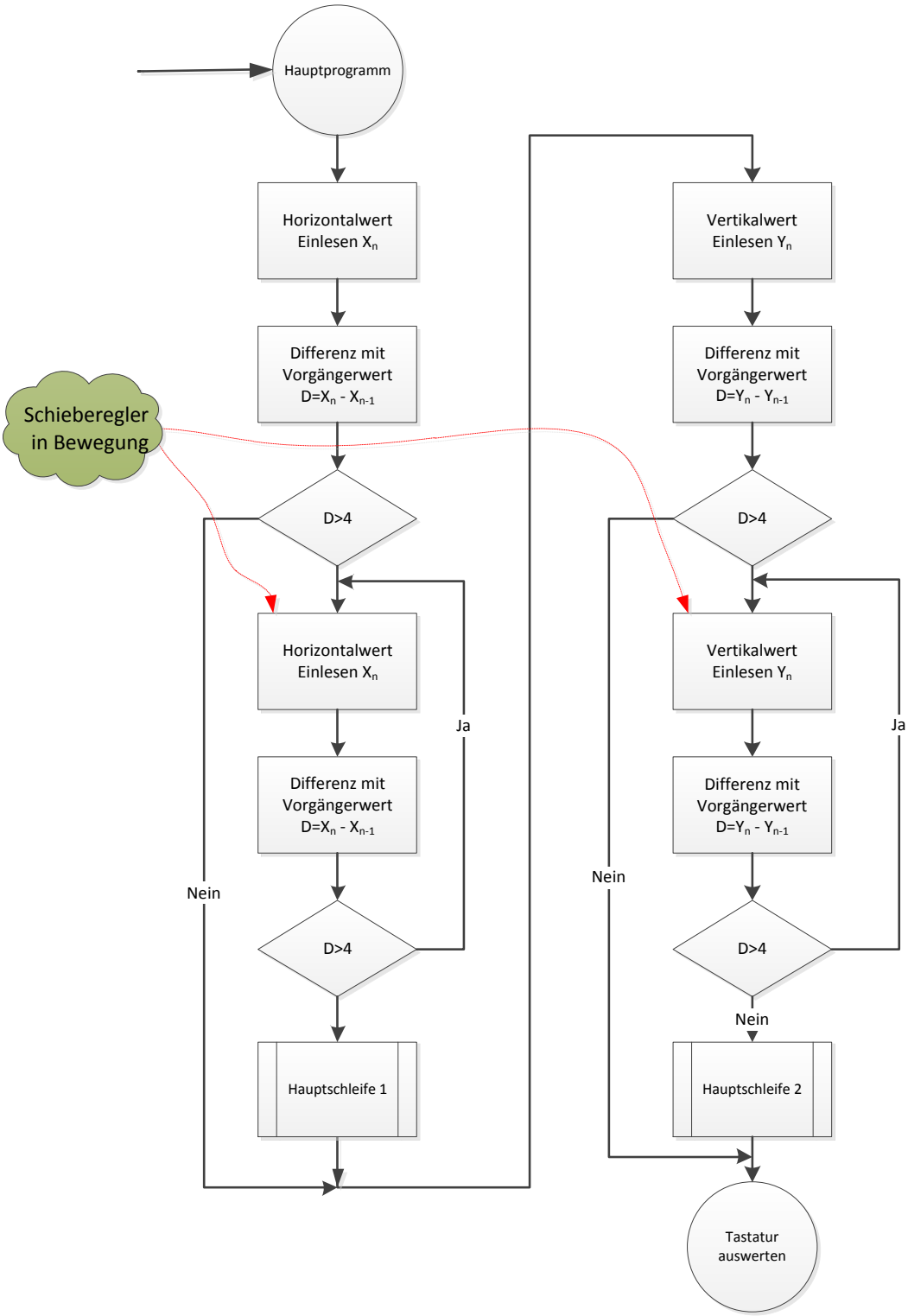


Abbildung 5-33: Flussdiagramm Hauptprogramm

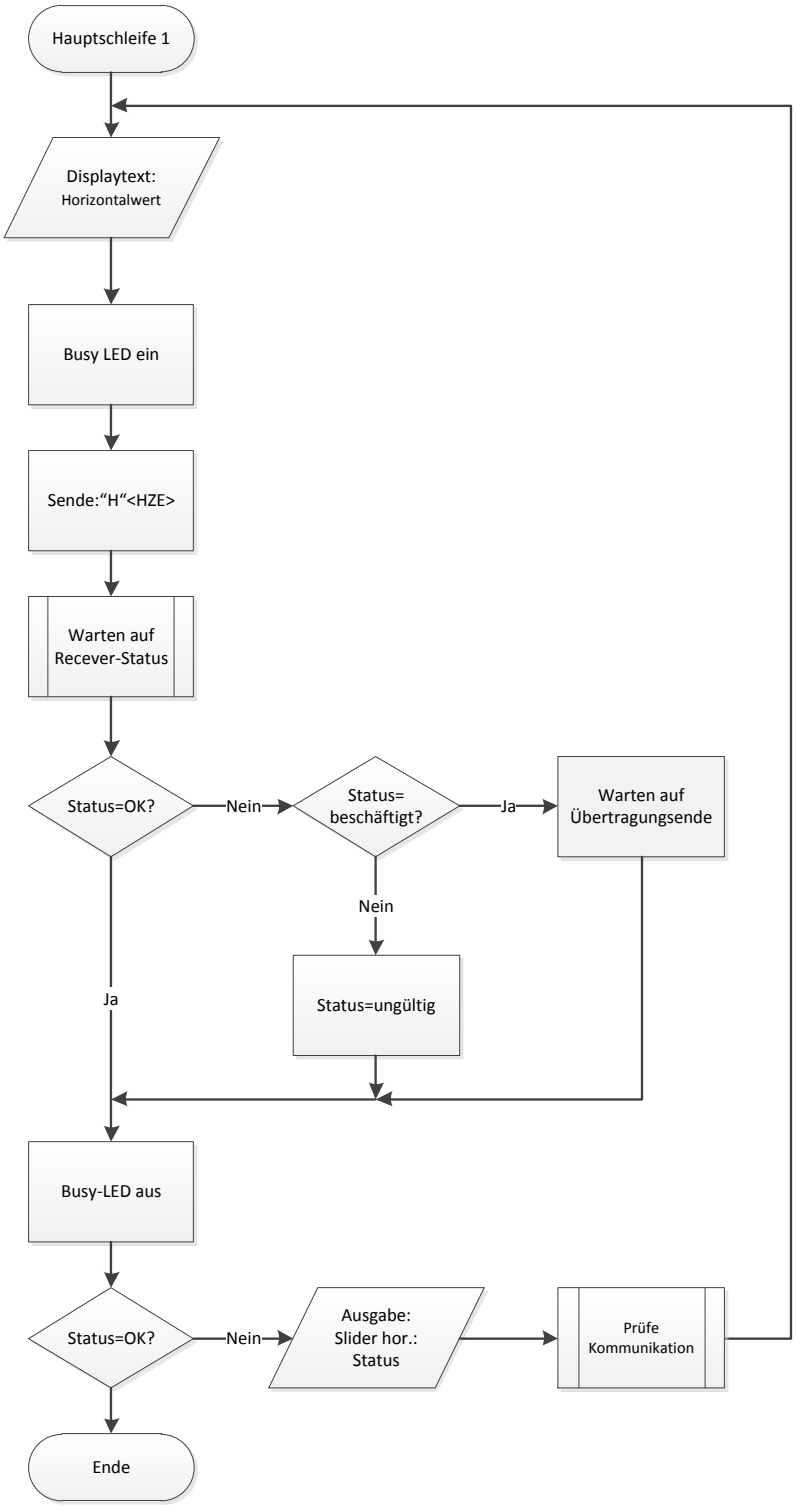


Abbildung 5-34: Flussdiagramm Hauptschleife 1

Das Flussdiagramm der Hauptschleife 2 ist identisch mit dem in Abbildung 5-34 dargestellten Flussdiagramm der Hauptschleife 1, es werden lediglich die Vertikal-Werte gesendet. Aus diesem Grund wird auf dessen Abbildung verzichtet.

Tastatur auswerten:

Die Tastatur wird eingelesen und ausgewertet. Wenn das Ergebnis im Intervall [1,…, 6] ist, das heißt ein Automatikprogramm ausgewählt wurde, muss dies dem Receiver mitgeteilt werden. Es wird der entsprechende Wert <1…6> gesendet und zum „*Procedure Mode“* gewechselt. Darauf erfolgt der Unterprogrammaufruf „*Drive Mode*“ mit der Parameterübergabe *Schrittweite* und *Schrittanzahl*. Im „*Drive Mode*“ wird das Ablaufprogramm schrittweise abgearbeitet.

Übergabeparameter für die einzelnen Tasten:

Taste <1> =>	5 Grad Schrittweite; Schrittanzahl 72
Taste <2> =>	10 Grad Schrittweite; Schrittanzahl 36
Taste <3> =>	20 Grad Schrittweite; Schrittanzahl 18
Taste <4> =>	30 Grad Schrittweite; Schrittanzahl 12
Taste <5> =>	40 Grad Schrittweite; Schrittanzahl 9
Taste <6> =>	72 Grad Schrittweite; Schrittanzahl 5

Mit der Taste <9> wird mit „*Display Procedures*“ fortgefahren. In diesem Programmteil werden die möglichen Automatikprogramme mit Schrittweite und Schrittanzahl am Display angezeigt. Der Bediener wird über eine Displayausgabe aufgefordert eine Taste <1…6> zu drücken. Wird eine entsprechende Taste gedrückt, wird am Display die dem Ablaufprogramm zugeordnete Schrittweite und Schrittanzahl angezeigt. Mit der Taste <*> erfolgt der Abbruch von „*Display Procedures*“. Diese Anzeigefunktion macht den T^2Controller zu einem selbsterklärenden Gerät, denn der Bediener muss sich die vorgegebenen Automatikprogramme nicht merken. Die entsprechenden Eingabemöglichkeiten werden immer am Display dargestellt.

Mit der Taste <0> wird mit „*Relay On*“ fortgefahren. Diese Funktion dient zum Auslösen eines Schaltkanals (Kameraauslösung). Es muss der Wert „0“ an den Receiver gesendet werden um die Kamera auszulösen.

In Abbildung 5-35 ist das Flussdiagramm für die Tastaturauswertung abgebildet.

Die Abbildungen der restlichen Flussdiagramme sind im Anhang B zu finden.

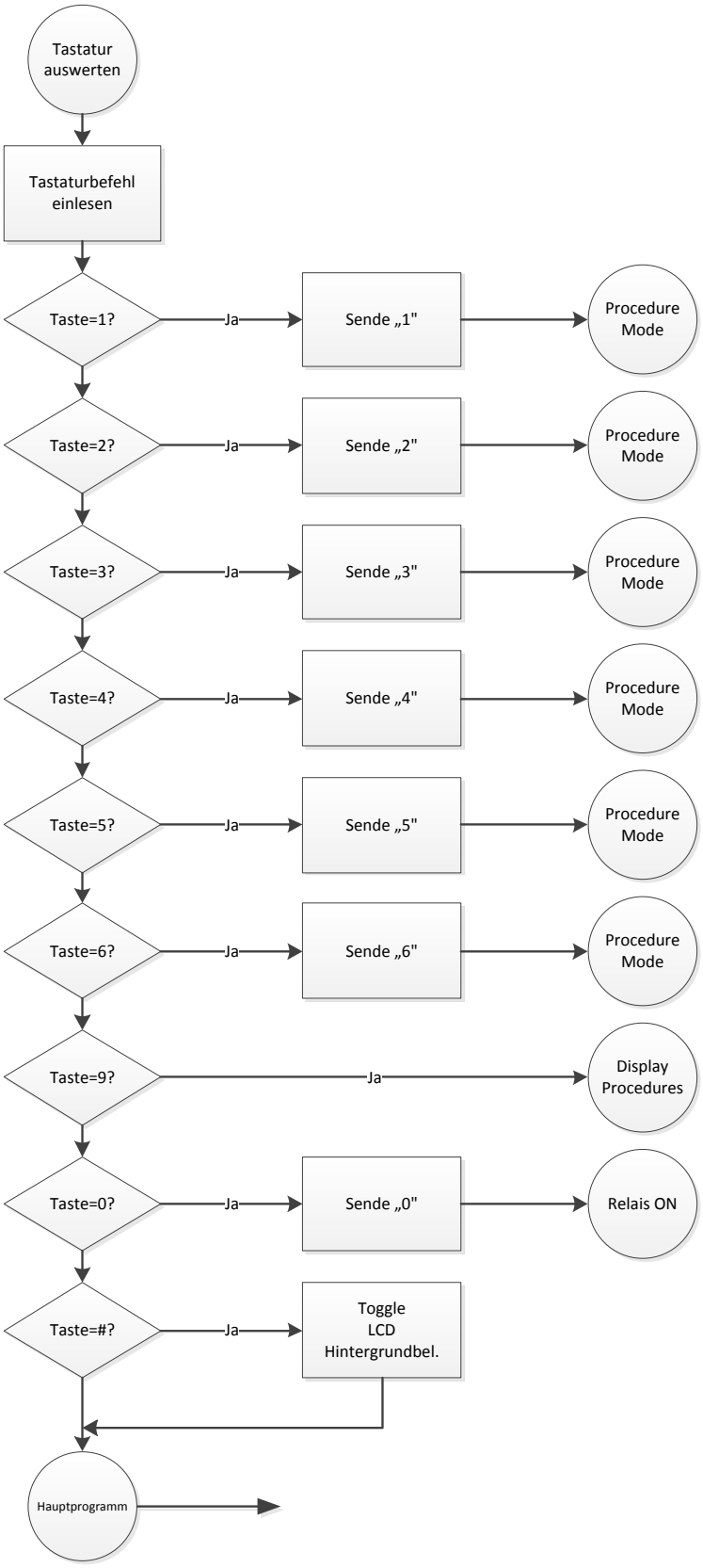


Abbildung 5-35: Flussdiagramm Tastaturauswertung

5.2 Softwarestruktur

Die Softwareerstellung erfolgt modular. Aufbauend auf die Grundfunktionalitäten wurde mit den Ablaufdiagrammen der Quellcode erstellt.

5.2.1 Programm Header

Beim Programm Header handelt es sich um einen allgemeinen Dokumentationsteil, in diesem soll das Softwareprojekt überblicksmäßig beschrieben, die Portbelegungen des Mikrocontrollers dokumentiert, sowie Statusmeldungen und Befehle aufgelistet werden.

Auszug aus dem Quellcode:

```
'------------------------------------------------------------------------------
' Project:      t2control Turn-Table control
' File:         t2c_commander-main.bas
' Date:         2011
' Author:       Christoph Köhle <koehle@htlinn.ac.at>
' Company:      NeCON
'
' Micro:        ATmega32@10MHz
'
' Hardware:     t2cCommander <t2Control>
'
' Description: Hauptschleife der Firmware t2control Commander.
'
' Ports:        PortA.0 -> Analog horizontal
'               PortA.1 -> Analog vertikal
'               PortA.7 -> Extern (fuer Erweiterung)
'               PortB.0 -> 3x4 Tastatur Spalte 1
'               PortB.1 -> 3x4 Tastatur Spalte 2
'               PortB.2 -> 3x4 Tastatur Spalte 3
'               PortB.3
'               PortB.4 -> 3x4 Tastatur Zeile 1
'               PortB.5 -> 3x4 Tastatur Zeile 2
'               PortB.6 -> 3x4 Tastatur Zeile 3
'               PortB.7 -> 3x4 Tastatur Zeile 4
'               PortC.0 -> LC-Display R/S
'               PortC.1 -> LC-Display Enable
'               PortC.2 -> Status-Led
'               PortC.3 -> Busy-Led
'               PortD.0 -> RxD
'               PortD.1 -> TxD
'               PortD.2 ->
'               PortD.3 -> LC-Display Hintergrundbeleuchtung
'               PortD.4 -> LC-Display D4
'               PortD.5 -> LC-Display D5
'               PortD.6 -> LC-Display D6
'               PortD.7 -> LC-Display D7
'------------------------------------------------------------------------------
```

Der gesamte Programm-Header ist dem im Anhang beigefügtem Quellcode zu entnehmen.

5.2.2 Deklarationsteil

Im Deklarationsteil werden die Variablen, Konstanten sowie Unterprogramme und Funktionen deklariert.

Auszug aus dem Quellcode:

```
'*****************************************************************************
'*  SUB ROUTINES
'*****************************************************************************
Declare Sub Display_status
Declare Sub Display_slider
Declare Sub Display_text
Declare Sub Display_lowerline
Declare Sub Check_communication
Declare Sub Display_procedure(angle As Byte , Step_size As Byte)

Declare Function Receive_char(byval Timeout As Integer) As Byte
Declare Function Receiver_synchronisation() As Byte
Declare Function Get_adw(byval Channel As Byte) As Integer
Declare Function Drive_mode(angle As Byte , Step_size As Byte) As Byte


'*****************************************************************************
'*  FLAGS
'*****************************************************************************
Const True = 1
Const False = 0


'*****************************************************************************
'*  CONSTANTS
'*****************************************************************************
Const Timer1_reload = 4880                                  'Timer1 Ladewert: 05s@10MHz
Const Synch_char = &B01010101
Const Busy_char = &B11110000
Const Shooton_char = &B11110001
Const Shootoff_char = &B11110010
Const Escape_char = &B11110100

Const Ok = 0

Const Synch_timeout = 8
Const Invalid_synch_char = 9
Const Invalid_pos_value = 20
Const Rs232_timeout = 100
Const No_valid_busy_char = 101
Const No_valid_shooton_char = 102
Const No_valid_shootoff_char = 103
Const No_valid_escape_char = 104

Const Receive_timeout_1 = 1000
Const Receive_timeout_2 = 10000
```

5.2.3 INIT - Teil

Im INIT – Block werden die verwendeten Peripherie – Komponenten wie:

- I/O Pins
- Keyboard
- Serial Ports
- Timer
- Analog – Digital – Converter
- LC – Display

initialisiert.

Auszug aus dem Quellcode:

```
'*******************************************************************************
'*  I/O PINs
'*******************************************************************************
Config PINC.2 = Output
Statusled Alias PORTC.2
Statusled = 1

Config PINC.3 = Output
Busyled Alias PORTC.3
Busyled = 1

Config PIND.3 = Output
Lcdbacklight Alias PORTD.3
Lcdbacklight = 1


'*******************************************************************************
'*  KEYBOARD CONFIGURATION
'*******************************************************************************
Config Kbd = PORTB , Delay = 100


'*******************************************************************************
'*  SERIAL PORTS CONFIGURATION
'*******************************************************************************
Config Serialin = Buffered , Size = 10

Config Com1 = 9600 , Synchrone = 0 , Parity = None , Stopbits = 1 , Databits = 8 ,
Clockpol = 0


'*******************************************************************************
'*  TIMER1 CONFIGURATION
'*******************************************************************************
Config Timer1 = Timer , Prescale = 1024                    'Timer1 Moder overflow
Enable Timer1
On Timer1 Isr_timer1                                      'Timer1 ISR-Aufruf
Load Timer1 , Timer1_reload


'*******************************************************************************
'*  ADC CONFIGURATION
'*******************************************************************************
Config ADC = Single , Prescaler = 16 , Reference = Internal
Start ADC                                                 'ADC einschalten
```

5.2.4 Code – Teil

Im Code – Teil befinden sich die in den Flussdiagrammen beschriebenen Programmmodule (Hauptschleife, Funktionen, Prozeduren und Tabellen) umgesetzt in BASCOM-Code.

Das Programm besteht aus folgenden Modulen:

- Startmeldung
- Mit Receiver verbinden
- Slider horizontal und vertikal auf Nullstellung prüfen
- Slider horizontal
- Slider vertikal
- Tastatur

Prozeduren und Funktionen:

- Display Status
- Display Slider
- Display Text
- Display Lowerline
- Receive Character
- Receiver Synchronisation
- Check Communikation
- Get ADW
- Drive Mode
- Display Procedure

Auszug aus dem Quellcode:

```
'++++++++++++++++++++++
'Mit Receiver verbinden
'++++++++++++++++++++++
Row1 = "Mit Receiver"
Row2 = "verbinden"
Display_text
Status = Receiver_synchronisation()
Stop Timer1                                              'Timer1 Overflow stoppen
Select Case Status
   Case Synch_timeout : Statusled = 1                    'Status-LED aus
                        Row1 = "Kein Receiver"
                        Row2 = "gefunden!"
                        Display_text
                        End
   Case Invalid_synch_char : Statusled = 1               'Status-LED aus
                             Row1 = "Synchronisation"
                             Row2 = "fehlgeschlagen!"
                             Display_text
                             End                         'Programm stoppen
End Select
Statusled = 0                                            'Status-Led ein
```

Der gesamte Quellcode befindet sich im Anhang B (Teil: Software), deshalb wird an dieser Stelle auf die Darstellung verzichtet.

5.3 Beschreibung der Funktionalität

Nach dem Einschalten des Controllers erfolgt die Anzeige einer Startmeldung am Display, für ca. 3 Sekunden. Der Controller versucht sich mit dem Receiver zu verbinden. Dies wird dem Benutzer wieder am Display mitgeteilt. Wenn der Verbindungsaufbau nicht funktioniert hat, wird am Display ausgegeben: „Kein Receiver gefunden“ bzw. „Synchronisation fehlgeschlagen“.

Nach erfolgreichem Verbindungsaufbau werden die Slider – Positionen geprüft. Diese müssen in Nullposition stehen um eine unerwünschte Bewegung der Kameraplattform zu verhindern. Sollten die Schieberegler nicht in Nullposition stehen, wird

der Benutzer mittels Displayausgabe aufgefordert die Slider in Nullposition zu bringen.

Nach erfolgter Nullpositionierung ist der Controller betriebsbereit. Die Status – LED leuchtet. Bei gestörter Verbindung mit dem Receiver blinkt die Status – LED.

Der Benutzer hat nun die Möglichkeit die Kameraplattform einerseits manuell über die Schieberegler zu steuern oder ein Ablaufprogramm über die Tastatur auszuwählen.

Bei der manuellen Betriebsart wird die gewünschte Horizontal – und Vertikalposition mittels Schieberegler (Slider) eingestellt, wobei die Einstellung am Display ersichtlich ist.

Das manuelle Auslösen der Kamera erfolgt über das Betätigen der Taste <0>.

Entscheidet sich der Benutzer für den Automatikmodus, so kann er zwischen 6 verschiedenen Ablaufprogrammen wählen. Die Auswahl erfolgt mit den Tasten <1…6>.
Auswahl der Ablaufprogramme mit Tasten <1…6>:

<1> Automatik-Modus horizontal 5 Grad Schrittweite ; Schrittanzahl 72
<2> Automatik-Modus horizontal 10 Grad Schrittweite; Schrittanzahl 36
<3> Automatik-Modus horizontal 20 Grad Schrittweite; Schrittanzahl 18
<4> Automatik-Modus horizontal 30 Grad Schrittweite; Schrittanzahl 12
<5> Automatik-Modus horizontal 40 Grad Schrittweite; Schrittanzahl 9
<6> Automatik-Modus horizontal 72 Grad Schrittweite; Schrittanzahl 5

Auch im Automatikmodus wird die Vertikalposition mit dem dafür vorgesehenen Schieberegler eingestellt und bleibt während der Ausführung eines Ablaufprogrammes unverändert.
Während der Abarbeitung der Automatikprogramme, wird am Display der Fortschritt laufend angezeigt. Der Automatikmodus kann mit der Taste <#> jederzeit gestoppt werden. Bei Verbindungsunterbrechung erfolgt eine Statusausgabe am Display.
Nach Beendigung des Ablaufprogrammes fährt die Kameraplattform in die Stellung der Sliderpositionen.

Mit der Taste <9> können die Details der Ablaufprogramme angezeigt werden. Nach Betätigung der Taste <9>, wird der Bediener aufgefordert mittels der Tasten <1…6> auszuwählen welches Programm angezeigt werden soll. Der Abbruch erfolgt mit der Taste <*>.
Die Taste <#>, dient dem Ein- bzw. Ausschalten der LCD – Hintergrundbeleuchtung.

6 Test des Systems

In diesem Kapitel wird der Testablauf der Hardware und Software beschrieben.

6.1 Test der Hardware

Nach der Fertigstellung der Hardware muss diese einiger Tests unterzogen werden, um sicherzustellen, dass alle Komponenten funktionieren und auch bei der Schaltungs- und Layout- Erstellung keine Fehler gemacht worden sind.

Nachdem die Schaltung überprüft worden ist und keine Fehler festgestellt werden konnten, wurde der *Board* – Befehl im Schaltplaneditor EAGLE ausgeführt und somit alle elektrischen Verbindungen in die Layout – Konstruktion übernommen. Der EAGLE – Befehl *Electrical Rule Check* prüft Schaltplanlogik und Konsistenz zwischen Schaltplan und Platine. *Design Rule Check* prüft z.B. die Platine auf Kurzschlüsse oder bestimmte Maße von Pads und Leiterbahnen.

Die EAGLE Board-Datei der fertigen Platine wurde an die Firma LeitOn gesendet und der Fertigungsauftrag erteilt. Vor Auftragsbestätigung der Firma LeitOn wurde das Layout einer nochmaligen Überprüfung unterzogen und nach telefonischer Rücksprache konnte die Prototypplatine gefertigt werden.

Die Hardwareprüfung des fertig bestückten Prototyps wurde wie folgt durchgeführt:

- Messung der Betriebsspannungen der Komponenten
- Abgleich der Spannungsteiler für Analogeingänge (R13 bzw. R14, siehe Abb. 4-21)
- Startmeldung auf Display ausgeben
- Tasten einlesen und auf Display ausgeben
- Hintergrundbeleuchtung des Displays ein- und ausschalten
- ADW-Wert der Schieberregler einlesen und den Wert am Display ausgeben
- Status – LED und Busy – LED ein- und ausschalten

Die Funktionalität der seriellen Schnittstelle bzw. das Funkmodul Wi.232 wird in Verbindung mit der Software getestet.

6.2 Test der Software

Vor dem eigentlichen Softwaretest musste noch die Kommunikationsschnittstelle (RS232 bzw. Wi.232) getestet werden.

Im ersten Schritt wurde der Controller mittels Pegelwandler – Interface (Fa. NeCON Abb. 6-36) mit dem Notebook verbunden. Die Verbindung wird vom Commander T^2 Control (Schnittstelle JP3, siehe Abb. 4-21) über ein Pegelwandler–Interface mit einem USB – RS232 - Umsetzer und USB – Anschluss der Notebooks realisiert. Am Notebook konnte nun das Programm Tera Term Version 4.68 (vgl./12/) gestartet und die Hardwareparameter (9600 Baud, 8 Datenbits, no parity und 1 Stopbit) der RS232 – Schnittstelle eingestellt werden. So konnte geprüft werden, ob die am Controller eingegebenen Befehle am Terminal ankommen.

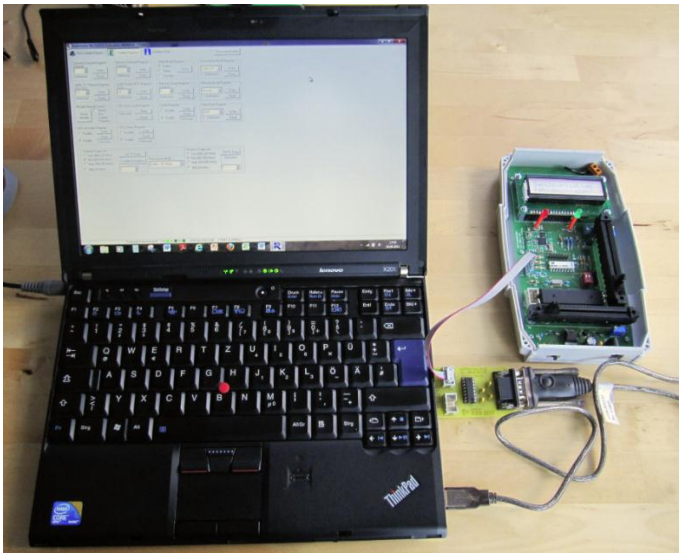


Abbildung 6-36: Verbindung zwischen Notebook und Commander

Im zweiten Schritt fungierte das Terminalsystem als Sender. Die vom Terminalprogramm gesendeten Daten konnten eingelesen, ausgewertet und der Status am Display ausgegeben werden.

Anschließend wurde die Funkstrecke mit dem Wi.232 – Modul aufgebaut und damit die Kabelverbindung ersetzt. Auf die Parametrierung des Wi.232 - Moduls wird später in diesem Kapitel noch genauer eingegangen. Alle vorher durchgeführten Testschritte wurden noch einmal, mittels Kommunikation über die Funkstrecke durchgeführt. Für das zweite Wi.232 – Modul zum Aufbau der Funkstrecke wurde das externe Funkmodul der derzeitigen Lösung verwendet.

Nachdem alle Tests erfolgreich waren, konnte das Notebook durch die Receiver - Hardware ersetzt werden um noch während der Programmierung die Programmabschnitte zu testen.

Insbesondere wurden Szenarien getestet, die die verschiedenen Statusmeldungen zurückgeben.

Statuscode	Bemerkung
0	OK
8	Timeout Receiver Synchronisation
9	Kein gültiges Synchronisationszeichen
20	Kein gültiger Positionswert empfangen
100	Timeout Empfang serieller Daten
101	Kein gültiges Receiver Busy-Zeichen empfangen
102	Kein gültiges Receiver Shoot-On-Zeichen empfangen
103	Kein gültiges Receiver Shoot-Off -Zeichen empfangen
104	Kein gültiges Receiver Escape-Zeichen empfangen

Tabelle 6-5: Statuscodes

Diese Tests wurden in Zusammenhang mit der Kameraplattform (Receiver-Hardware) durchgeführt um reale Verhältnisse zu simulieren.

Als Beispiel sei angeführt:

- Reichweitentest (Statuscode 100)
- Unterbrechung der Spannungsversorgung an der Receiver-Hardware (Statuscode 8, 20, 100-104)

Parametrierung des Wi.232 Modules:

Zur Parametrierung des Funkmodules Wi.232 wird auf der Webseite der Firma RADIOTRONIX (http://www.radiotronix.com/products/proddb.asp?ProdID=198) ein Softwaretool zum Download bereitgestellt. Nach Installation des Tools kann der Commander über die vorgesehene Schnittstelle JP5 (siehe Abb. 4-21) mit dem Notebook verbunden werden. Dabei ist zu beachten, dass die Spannungsversorgung des Commanders vorher eingeschaltet wird, da sonst der Wi.232 auf die Default – Werte zurückgesetzt wird. Der Schalter S1 (siehe Abb. 4-21) muss ebenfalls umgelegt werden um die Programmierschnittstelle zu aktivieren.

In der folgenden Abbildung sind jene Einstellungen gekennzeichnet die vorzunehmen sind:

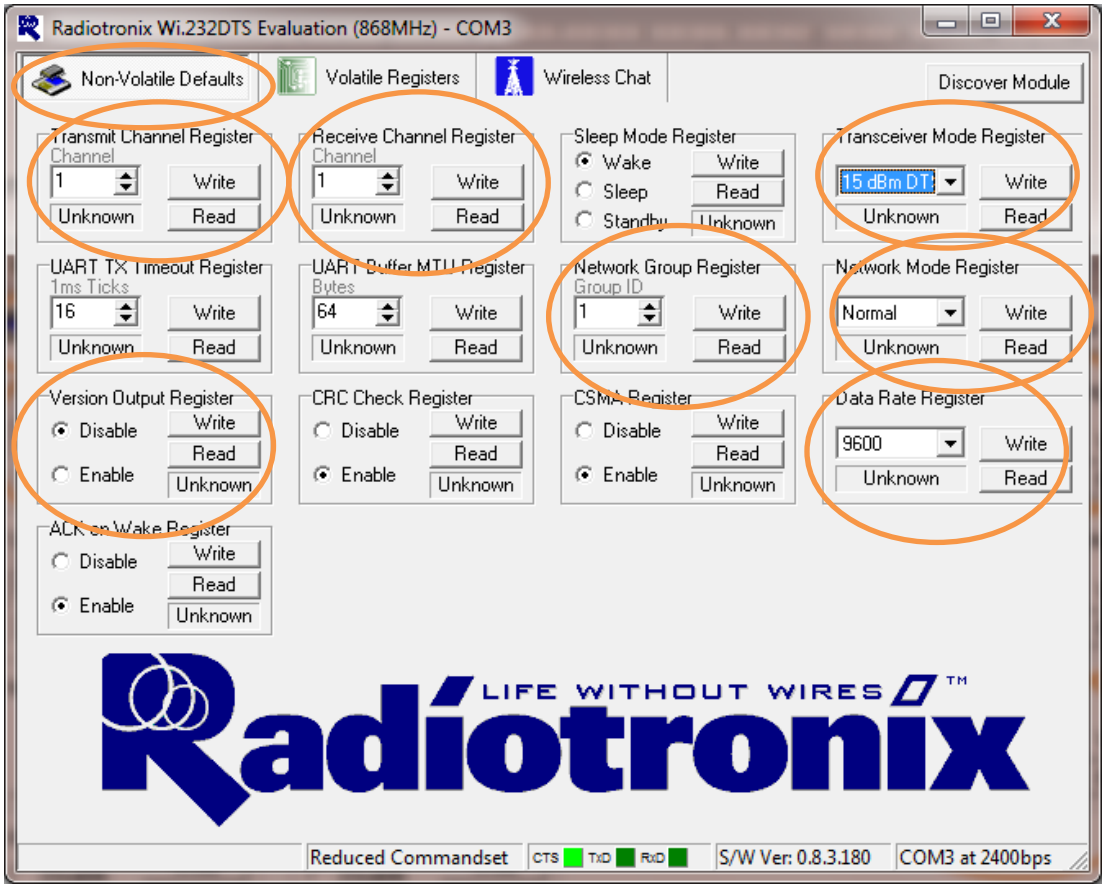


Abbildung 6-37: Wi.232 Evaluation - Software – Eingabefenster

Mit WRITE werden die Werte übernommen und mit READ können die aktuellen Einstellungen des Wi.232 – Moduls ausgelesen werden.

6.3 Abschlusstest im realen Feldeinsatz

Beim abschließenden Test soll erprobt werden, wie sich der Commander im praktischen Einsatz bewährt.
Dabei soll eine Überprüfung aller Elektroniksysteme (Drohne, Telemetrie-Datenübertragung und Kamerasteuerung) durchgeführt werden, um eventuelle Konflikte zwischen den Systemen festzustellen.

Diese Tests wurden gemeinsam mit der Firma NeCON durchgeführt, da diese mit der Bedienung der Flugdrohne und mit der technischen Vermessung vertraut sind.

Die gewonnenen Erkenntnisse aus den vorausgegangenen Tests werden im Kapitel 7 unter Ergebnisse zusammengefasst.

7 Zusammenfassung

Im abschließenden Kapitel werden die bisher gewonnenen Ergebnisse zusammengefasst und eine Bewertung der Leistung aus Sicht des Autors vorgenommen. Ein Ausblick zeigt Weiterentwicklungspotenziale auf.

7.1 Ergebnisse

Der in dieser Diplomarbeit entwickelte Prototyp erfüllt alle in der Aufgabenstellung geforderten Funktionalitäten. Beim Hardwareaufbau wurde darauf geachtet, Komponenten zu verbauen, welche sicher noch in Zukunft auf dem Markt erhältlich sind und somit das Kriterium der Serientauglichkeit erfüllt. Das ausgewählte Gehäuse erwies sich als ausgesprochen vorteilhaft, was die Wartungs- und Bedienerfreundlichkeit betrifft. Anordnung von Display, Tastatur und Schieberegler machen den Controller nicht nur optisch sondern auch funktionell zu einem ansprechenden Gerät.

Beim Praxistest des Controllers kam es sehr häufig zur Unterbrechung der Kommunikationsverbindung. Im Sichtbereich bis ca. 90 Meter Entfernung funktioniert die Kommunikation noch störungsfrei, aber mit zunehmender Entfernung nimmt auch die Fehlerhäufigkeit (Datenübertragungsausfälle) zu. Die Ursache dafür muss beim Controller liegen, da dieser Fehler früher, bei der Steuerung über Notebook und Sendemodul, nie aufgetreten ist. Eine mögliche Ursache könnte die Printantenne sein. Für eine genaue Antennenanpassung wird die Umrüstung des Controllers auf eine externe, kommerzielle Antenne in Betracht gezogen.

Ein weiteres Problem stellt die Elektromechanik der Kameraplattform dar, denn diese lässt sich nicht genau 360° bewegen. Um dieses Problem zu lösen, ist eine Anpassung der Software auf die Anfangs- und Endposition des Kameragestells erforderlich (Trimmung des Systems).

Das Auslösen der Kamera über die Folientastatur erwies sich ebenfalls als nicht so günstig. Eine eigene Auslösetaste könnte den Bedienungskomfort erheblich steigern.

7.2 Bewertung der Arbeit

Sowohl Hard- als auch Software sind in der derzeitigen Ausführung für den gedachten Einsatzbereich tauglich. Bevor der Controller jedoch in Serie gefertigt werden kann, sind geringfügige Änderungen der Hardware nötig (externe Antenne). Die Änderungen der Software sind dann jederzeit möglich, wobei dies ein laufender Prozess sein wird. Auf jeden Fall bildet der in dieser Arbeit entwickelte Prototyp die Grundlage für das Serienprodukt und alle daraus resultierenden Ergebnisse können genutzt werden.

7.3 Ausblick

Der Commander T^{2}Control wird nach Überarbeitung der Hardware und Ergänzung der Software in fünffacher Ausführung gefertigt und für die Abwicklung anstehender Aufträge der Firma NeCON eingesetzt.

Die Änderungen umfassen:

- Umrüstung auf eine externe Antenne. Dazu sind folgende Bauteile nötig:

Bezeichnung	Bestell-Nr.
Koax-Kabel SMA-MMCX 6"	2509565991
MMCX Vertikal Buchse	713-0971
Antenne SMA - 868	542-551

Tabelle 7-6: Bauteile für Umrüstung (RS – Components)

- Trimmung der Anfangs- und Endposition des Kameragestells über Software

Was das Auslösen und die Bewegung der Kamera angeht, wird überlegt, die Schieberegler durch einen Joystick mit Druckknopf zu ersetzen und somit den Bedienungskomfort zu erhöhen. Diese Änderung wird aber noch nicht umgesetzt, da zuerst in der Praxis die nötigen Erfahrungen gesammelt werden sollen.

Literatur

/1/ Programmieren der AVR RISC Mikrocontroller, Claus Kühnel, 3.Aufl. ISBN 978-3-907957-14-4

/2/ Halbleiter Schaltungstechnik, U.Tietze/ Ch. Schenk, Springer Verlag 2002, 12. Aufl. ISBN 3-540-42849-6

/3/ URL: http://at.rs-online.com (Verfügbar am 02.07.2011) Bauteilinformationen zu Tabelle 4-3

/4/ URL: http://www.cadsoft.de/ (Verfügbar am 14.07.2011) Informationen zu EAGLE

/5/ EAGLE Handbuch Version 5, 8.Aufl , erhältlich als PDF unter /4/

/6/ URL: http://www.alldatasheet.com/ (Verfügbar am 14.07.2011) Informationen zu Spannungsreglern

/7/ URL: http://www.atmel.com/products/AVR/ (Verfügbar am 14.07.2011) Informationen zu ATMega 32

/8/ Beierlein, Thomas; Hagenbruch, Olaf: Mikroprozessortechnik. 3. Auflage. Leipzig: Carl Hanser Verlag. 2004

/9/ Nachrichtentechnik, E. Herter / W. Lörcher, Hanser Verlag, 1990, 5. Aufl., ISBN 3-446-15964-9

/10/ URL: http://www.radiotronix.com/datasheets/new/eur_um.pdf (Verfügbar am 16.07.2011) Informationen zu Wi.232

/11/ URL: http://sprut.de/electronic/lcd/ (Verfügbar am 17.07.2011) Informationen zu Punktmatrix-Anzeigen

/12/ URL: http://ttssh2.sourceforge.jp (Verfügbar am 28.07.2011)

/13/ Einfacher Einstieg in die Elektronik mit AVR – Mikrocontroller und BASCOM, Stefan Hoffmann, Books und Demand GmbH, 1.Aufl. 2010, ISBN 978-3-8391-8430-1

/14/ URL: http://www.conrad.at/ce/de/ (Verfügbar am 17.07.2011) Bauteilinformationen zu Tabelle 4-3

/15/ URL: http://at.farnell.com (Verfügbar am 17.07.2011) Bauteilinformationen zu Tabelle 4-3

/16/ URL: https://www.distrelec.at (Verfügbar am 17.07.2011) Bauteilinformationen zu Tabelle 4-3

/17/ URL: http://www.mikrokopter.com (Verfügbar am 30.09.2011)

/18/ URL: http://www.rn-wissen.de/index.php/Bascom (Verfügbar am 30.09.2011) Informationen zu BASCOM

Anlagen

Im folgenden Abschnitt befinden sich alle wichtigen Ergänzungen, die nötig sind den Projektverlauf nachzuvollziehen.

A: Anlagen Hardware

Bauteillisten:

Conrad

Bezeichnung	Bestell-Nr.	VP-Einheit	Benötigt
LCD Rahmen 2x16	142018 - 05	1	1
Textdisplay STN grau 2x16	181664 - 05	1	1
Schieberegler linear 470 Ohm	441414 - 05	1	2
Folientastatur Matrix 3x4	709930 - 05	1	1
Messerleiste 6polig	742499 - 05	1	1
Emmerich Akkupack 6V / 800mA	255054 - 05	1	1
Transistor BC338	154962 - 05	1	1
Ladegerät NiMH 300mA	250125 - 05	1	1
Widerstand 100 Ohm	403920 - 05	100	2
Widerstand 470 Ohm	404004 - 05	100	5
Widerstand 1k	404047 - 05	100	1
Widerstand 10k	404160 - 05	100	1
Schraube M2,5 x 20mm	839656 - 05	100	4

Distrelec

Bezeichnung	Bestell-Nr.	VP-Einheit	Benötigt
Schiebereglerknopf schwarz	260116	1	2
LED 5 mm rot	251538	1	1
LED 5 mm grün	251542	1	1
ATMEGA32 DIL-40	645110	1	1

RS Components

Bezeichnung	Bestell-Nr.	VP-Einheit	Benötigt
Schiebeschalter 1 Wechsler 90° gew.	429-482	1	1
Gehäuse Serie 33	528-882	1	1
Printklemme RM 5,08 2 polig	408-7871	5	1
IC Sockel 40 polig	674-2505	10	1
Quarz HC49S 10 MHz	226-1746	1	1
Trimmerpoti 1k	522-0063	1	2
Trimmpoti 10k	125-871	1	1
Abstandshalter 12,7 mm	606-793	50	4
Widerstandsnetzwerk 470 Ohm	625-3963	5	1
Spannungsregler LV50CV 5V / 0,5A	355-4257	1	1
Spannungsregler LP2950ACZ -3-3	536-0492	5	1
DIP Switsch 2 polig	712-2570	1	1
Kondensator 22 pF	652-9866	25	2
Kondensator 100n	721-5240	50	3
Elko 220 uF	715-2559	10	1
Tantal-Elko 2,2 uF	684-4367	5	2
Z-Diode 3,3V	812-336	5	1
Diode 1N4004	628-9029	10	2
Diode 1N4148	544-3480	20	1

Farnell

Bezeichnung	Bestell-Nr.	VP-Einheit	Benötigt
WI.232EUR-R	1565157	1	1

Datenblätter:

Die Datenblätter der verwendeten Komponenten sind unter dem angegebenen Link abrufbar.

Gehäuse:

(Quelle: http://at.rs-online.com/web/p/products/0528882/ verfügbar am 02.07.2011)

Folientastatur:

(Quelle: http://www.conrad.at/ce/de/product/709930/ verfügbar am 02.07.2011)

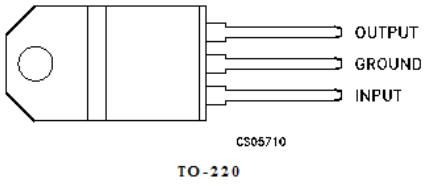
Schieberegler:

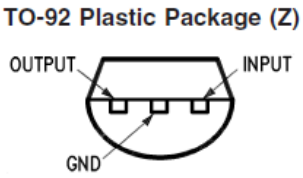
http://www.produktinfo.conrad.com/datenblaetter/425000-449999/441414-da-01-en-Mono_Schiebe_Potentiometer_CIPGST_2_58M.pdf

LC – Display:

http://www.produktinfo.conrad.com/datenblaetter/175000-199999/181664-da-01-en-LCD_MODUL_STN_POSITIV_LED_WEISS_16X2.pdf

Spannungsregler:





Gehäuse der Spannungsregler LF50CV und LP2950ACZ-3.3
(Quelle: http://www.alldatasheet.com/ verfügbar am 14.07.2011)

Controller:

http://www.atmel.com/dyn/resources/prod_documents/doc2503.pdf

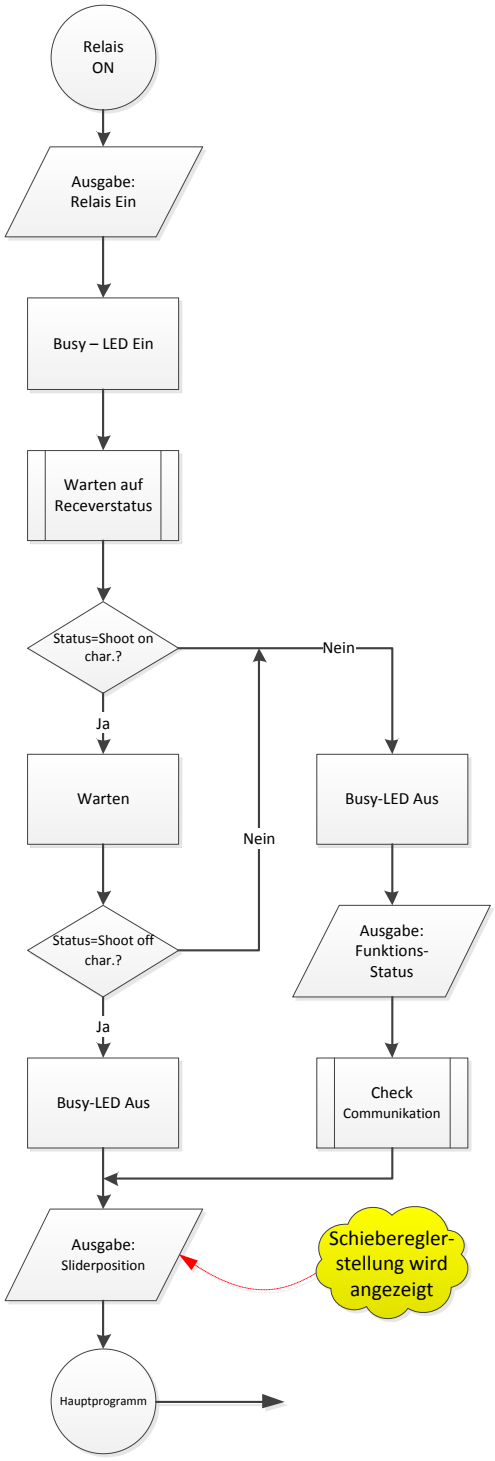
Wi.232:

http://www.radiotronix.com/datasheets/new/eur_um.pdf

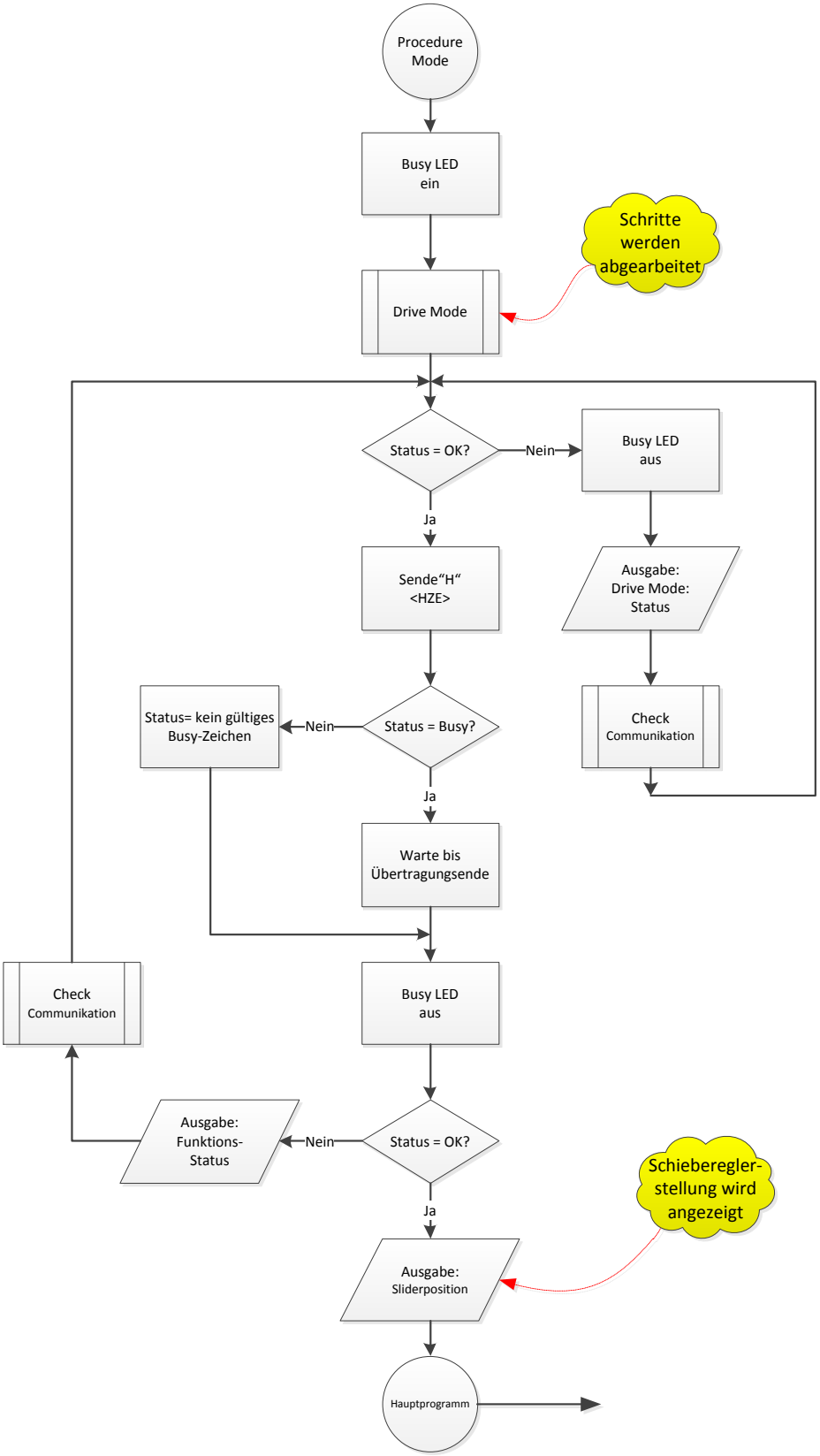
B: Anlagen Software

Flussdiagramme:

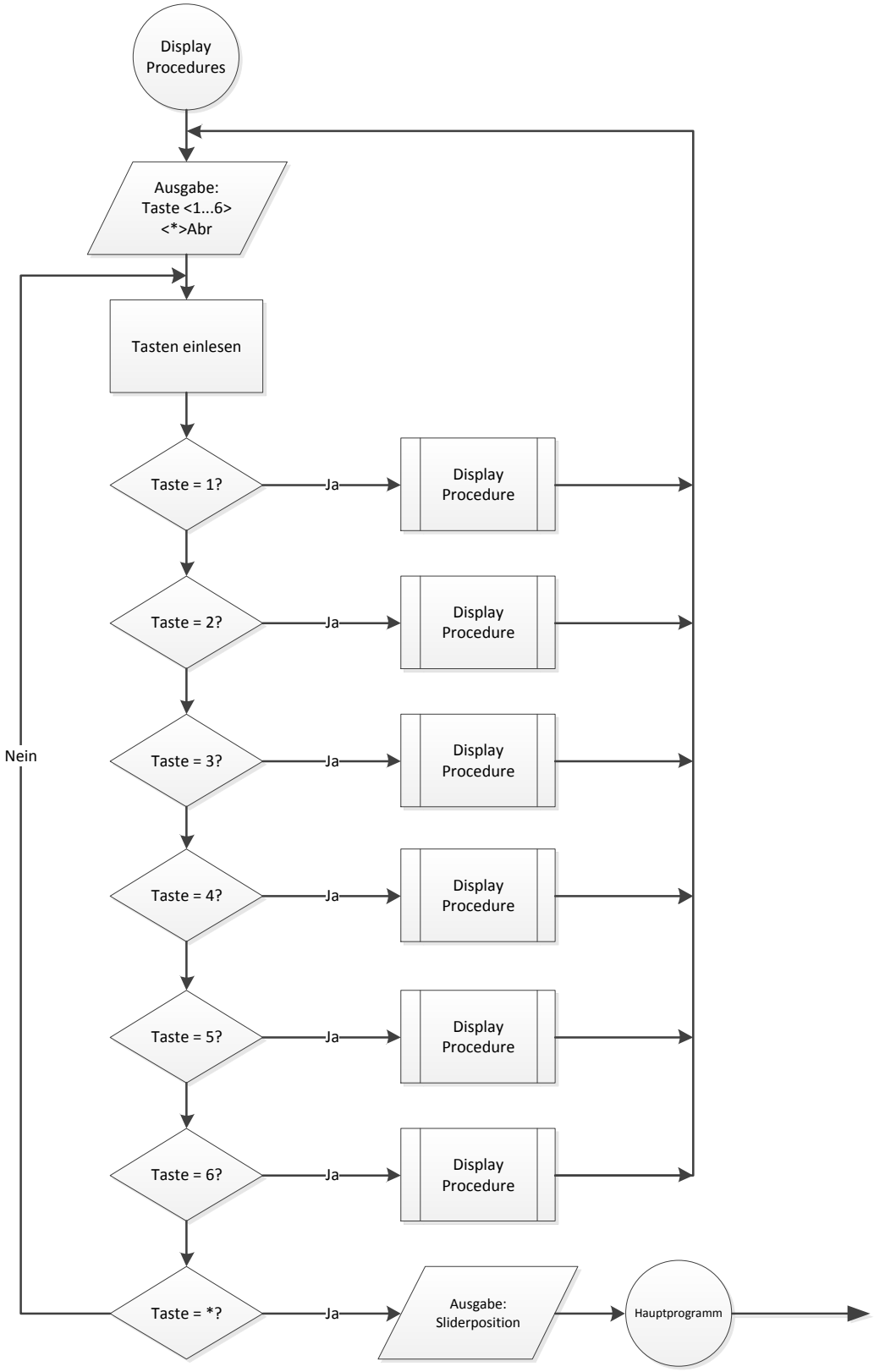
Flussdiagramm Relais ON:



Flussdiagramm Procedure – Mode:



Flussdiagramm Display Procedure:



Quellcode:

```
'-------------------------------------------------------------------------------
' Project:      t2control Turn-Table control
' File:         t2c_commander-main.bas
' Date:         2011
' Author:       Christoph Köhle <koehle@htlinn.ac.at>
' Company:      NeCON
'
' Micro:        ATmega32@10MHz
'
' Hardware:     t2cCommander <t2Control>
'
' Description: Hauptschleife der Firmware t2control Commander.
'
' Ports:        PortA.0 -> Analog horizontal
'               PortA.1 -> Analog vertikal
'               PortA.7 -> Extern (fuer Erweiterung)
'               PortB.0 -> 3x4 Tastatur Spalte 1
'               PortB.1 -> 3x4 Tastatur Spalte 2
'               PortB.2 -> 3x4 Tastatur Spalte 3
'               PortB.3
'               PortB.4 -> 3x4 Tastatur Zeile 1
'               PortB.5 -> 3x4 Tastatur Zeile 2
'               PortB.6 -> 3x4 Tastatur Zeile 3
'               PortB.7 -> 3x4 Tastatur Zeile 4
'               PortC.0 -> LC-Display R/S
'               PortC.1 -> LC-Display Enable
'               PortC.2 -> Status-Led
'               PortC.3 -> Busy-Led
'               PortD.0 -> RxD
'               PortD.1 -> TxD
'               PortD.2 ->
'               PortD.3 -> LC-Display Hintergrundbeleuchtung
'               PortD.4 -> LC-Display D4
'               PortD.5 -> LC-Display D5
'               PortD.6 -> LC-Display D6
'               PortD.7 -> LC-Display D7
'-------------------------------------------------------------------------------


'-------------------------------------------------------------------------------
' Tastaturbefehle:
'
' 1               Automatik-Modus horizontal 5 Grad Schrittweite
' 2               Automatik-Modus horizontal 10 Grad Schrittweite
' 3               Automatik-Modus horizontal 20 Grad Schrittweite
' 4               Automatik-Modus horizontal 30 Grad Schrittweite
' 5               Automatik-Modus horizontal 40 Grad Schrittweite
' 6               Automatik-Modus horizontal 72 Grad Schrittweite
' 7               Reserve
' 8               Reserve
' 9               Abfrage Ablaufdiagramme
' 9 -> 1          Abfrage Ablaufdiagramm #1
' 9 -> 2          Abfrage Ablaufdiagramm #2
' 9 -> 3          Abfrage Ablaufdiagramm #3
' 9 -> 4          Abfrage Ablaufdiagramm #4
' 9 -> 5          Abfrage Ablaufdiagramm #5
' 9 -> 6          Abfrage Ablaufdiagramm #6
' 0               Relais aktivieren
' *               Abbruch (Ausstieg aus den Untermenues)
' #               LCD-Hintergrundbeleuchtung EIN / AUS
'-------------------------------------------------------------------------------
```

```
'-------------------------------------------------------------------------------
' Kommandos (An den Receiver):
'
' 1               Automatik-Modus horizontal 5 Grad Schrittweite
' 2               Automatik-Modus horizontal 10 Grad Schrittweite
' 3               Automatik-Modus horizontal 20 Grad Schrittweite
' 4               Automatik-Modus horizontal 30 Grad Schrittweite
' 5               Automatik-Modus horizontal 40 Grad Schrittweite
' 6               Automatik-Modus horizontal 72 Grad Schrittweite
' 9               Abfrage Ablaufprogramme
' H <HZE>         Horizontale Position manuell vorgeben [000 .. 720]
'                 000 -> Null-Position (home): 0 GRD
'                 720 -> Maximal-Position: 360 GRD
' V <HZE>         Vertikale Position manuell vorgeben   [000 .. 720]


'-------------------------------------------------------------------------------
' Status:
'
'   0             OK
'   1             Reserve
'   2             Reserve
'   3             Reserve
'   4             Reserve
'
'   8             Timeout Receiver Synchronistion
'   9             Kein gueltigens Synch-Zeichen empfangen
'
'  20             Kein gueltiger Position-Wert empfangen
'
' 100             Timeout Empfang serieller Daten
' 101             Kein gueltiges Receiver Busy-Zeichen empfangen
' 102             Kein gueltiges Receiver ShootON-Zeichen empfangen
' 103             Kein gueltiges Receiver ShootOFF-Zeichen empfangen
' 104             Kein gueltiges Receiver Escape-Zeichen empfangen
'-------------------------------------------------------------------------------


$regfile = "m32def.dat"                                     'ATmega32
$crystal = 10000000                                         '10MHz
$hwstack = 250                                              'Hardware stack
$swstack = 250                                              'Software stack
$framesize = 250                                            'Frame space


'*******************************************************************************
'*  SUB ROUTINES
'*******************************************************************************
Declare Sub Display_status
Declare Sub Display_slider
Declare Sub Display_text
Declare Sub Display_lowerline
Declare Sub Check_communication
Declare Sub Display_procedure(angle As Byte , Step_size As Byte)

Declare Function Receive_char(byval Timeout As Integer) As Byte
Declare Function Receiver_synchronisation() As Byte
Declare Function Get_adw(byval Channel As Byte) As Integer
Declare Function Drive_mode(angle As Byte , Step_size As Byte) As Byte


'*******************************************************************************
'*  FLAGS
'*******************************************************************************
Const True = 1
Const False = 0
```

```
'*******************************************************************************
'*  CONSTANTS
'*******************************************************************************
Const Timer1_reload = 4880                                  'Timer1 Ladewert: 05s@10MHz
Const Synch_char = &B01010101
Const Busy_char = &B11110000
Const Shooton_char = &B11110001
Const Shootoff_char = &B11110010
Const Escape_char = &B11110100

Const Ok = 0

Const Synch_timeout = 8
Const Invalid_synch_char = 9
Const Invalid_pos_value = 20
Const Rs232_timeout = 100
Const No_valid_busy_char = 101
Const No_valid_shooton_char = 102
Const No_valid_shootoff_char = 103
Const No_valid_escape_char = 104

Const Receive_timeout_1 = 1000
Const Receive_timeout_2 = 10000


'*******************************************************************************
'*  VARIABLES
'*******************************************************************************
Dim Status As Byte , Eeprom_status As Byte
Dim Adw As Integer
Dim Delta As Integer
Dim Serial_char As Byte

Dim H_previous As Integer
Dim V_previous As Integer

Dim Row1 As String * 16                                     'Display 1.Zeile
Dim Row2 As String * 16                                     'Display 2.Zeile
Dim Horizontal As String * 3
Dim Vertical As String * 3

Dim Angle_str As String * 2
Dim Step_size_str As String * 2
Dim Step_counter_str As String * 2

Dim I As Integer , J As Integer , K As Integer              'Zaehlvariablen
Dim Kbd As Integer
Dim Kbdresult As Byte

Dim B As Byte
Dim W(6) As Byte , S(6) As Byte                             'Winkel, Schritte


'*******************************************************************************
'*  I/O PINs
'*******************************************************************************
Config PINC.2 = Output
Statusled Alias PORTC.2
Statusled = 1

Config PINC.3 = Output
Busyled Alias PORTC.3
Busyled = 1

Config PIND.3 = Output
Lcdbacklight Alias PORTD.3
Lcdbacklight = 1


'*******************************************************************************
'*  KEYBOARD CONFIGURATION
'*******************************************************************************
Config Kbd = PORTB , Delay = 100
```

```
'*****************************************************************************
'*  SERIAL PORTS CONFIGURATION
'*****************************************************************************
Config Serialin = Buffered , Size = 10

Config Com1 = 9600 , Synchrone = 0 , Parity = None , Stopbits = 1 , Databits = 8 ,
Clockpol = 0


'*****************************************************************************
'*  TIMER1 CONFIGURATION
'*****************************************************************************
Config Timer1 = Timer , Prescale = 1024                    'Timer1 Moder overflow
Enable Timer1
On Timer1 Isr_timer1                                       'Timer1 ISR-Aufruf
Load Timer1 , Timer1_reload


'*****************************************************************************
'*  ADC CONFIGURATION
'*****************************************************************************
Config ADC = Single , Prescaler = 16 , Reference = Internal
Start ADC                                                  'ADC einschalten


'*****************************************************************************
'*  LCD DISPLAY CONFIGURATION
'*****************************************************************************

Config Lcdpin = Pin , Db4 = PORTD.4 , Db5 = PORTD.5 , Db6 = PORTD.6 , Db7 = PORTD.7
Config Lcdpin = Pin , Rs = PORTC.0 , E = PORTC.1
Config Lcd = 16 * 2                                        '16 Char, 2 Zeilen
Cls
Cursor Off Noblink                                         'Cursor aus und nicht
blinken


'*****************************************************************************
'*  INIT
'*****************************************************************************
Row1 = "    t2cCommander"
Row2 = "V 1.0 (c)2011 ck"
Display_text
Wait 3

Enable Interrupts

'++++++++++++++++++++++
'Mit Receiver verbinden
'++++++++++++++++++++++
Row1 = "Mit Receiver"
Row2 = "verbinden"
Display_text
Status = Receiver_synchronisation()
Stop Timer1                                                'Timer1 Overflow stoppen
Select Case Status
   Case Synch_timeout : Statusled = 1                      'Status-LED aus
                        Row1 = "Kein Receiver"
                        Row2 = "gefunden!"
                        Display_text
                        End
   Case Invalid_synch_char : Statusled = 1                 'Status-LED aus
                             Row1 = "Synchronisation"
                             Row2 = "fehlgeschlagen!"
                             Display_text
                             End                           'Programm stoppen
End Select
Statusled = 0                                              'Status-Led ein
```

```
'+++++++++++++++++++++++++++++++++++++++++++
'Slider hor. & ver. auf Nullstellung pruefen
'+++++++++++++++++++++++++++++++++++++++++++
Adw = Getadc(0)                                 'ADC am Beginn 2x einlesenAdw
Adw = Getadc(0)                                 'horizontaler Analogwert
Cls
If Adw > 0 Then
   Row1 = "Schieberegler in"
   Row2 = "Nullpos. bringen"
   Display_text
Else
   Adw = Getadc(1)                              'vertikaler Analogwert
   If Adw > 0 Then
      Row1 = "Schieberegler in"
      Row2 = "Nullpos. bringen"
      Display_text
   End If
End If
Do
   Adw = Getadc(0)                              'horizontaler Analogwert
   Horizontal = Str(adw)
   Waitms 50
Loop Until Adw = 0
Do
   Adw = Getadc(1)                              'vertikaler Analogwert
   Vertical = Str(adw)
   Waitms 50
Loop Until Adw = 0

Horizontal = Format(horizontal , "000")
Vertical = Format(vertical , "000")
Display_slider

H_previous = 0                                  'Hor. Vorgaengerwert
vorbelegen
V_previous = 0                                  'Ver. Vorgaengerwert
vorbelegen

W(1) = 5                                        '5 Grad -->
S(1) = 72                                       '72 Schritte


W(2) = 10                                       '10 Grad -->
S(2) = 36                                       '36 Schritte

W(3) = 20                                       '20 Grad -->
S(3) = 18                                       '18 Schritte

W(4) = 30                                       '30 Grad -->
S(4) = 12                                       '12 Schritte

W(5) = 40                                       '40 Grad -->
S(5) = 9                                        '9 Schritte

W(6) = 72                                       '72 Grad -->
S(6) = 5                                        '5 Schritte
```

```
'*******************************************************************************
'*  MAIN LOOP
'*******************************************************************************
Main_loop:
'++++++++++++++++
'Slider horizontal
'++++++++++++++++
   Adw = Get_adw(0)                                    'horz. Analogwert einlesen
   Waitms 100
   Delta = Adw - H_previous                            'Abweichung bilden
   Delta = Abs(delta)                                  'Betrag bilden
   If Delta > 4 Then                                    'Slider in Bewegung
      Busyled = 0                                      'Busy-Led ein
      Do                                               'Solange bis Slider in Ruhe
         H_previous = Adw                              'Aktuellen Wert retten
         Adw = Get_adw(0)
         Waitms 100
         Delta = Adw - H_previous
         Delta = Abs(delta)
      Loop Until Delta < 4
      Horizontal = Str(adw)
      Horizontal = Format(horizontal , "000")
Main_loop_1:
      Display_slider
      Print "H" ; Horizontal                           'Befehl an Receiver
      Status = Receive_char(receive timeout 1)         'Wait for Receiver Busy-Char.
      If Status = Busy_char Then
         Status = Receive_char(receive_timeout_2)      'Wait for Receiver Char.
      Else
         Status = No_valid_busy_char
      End If
      Busyled = 1                                      'Busy-Led wieder aus
      If Status > 0 Then
         Row1 = "Slider hor.:"
         Display_status
         Check_communication
         Goto Main_loop_1
      End If
   End If

   ___ __
'++++++++++++++++
'Slider vertikal
'++++++++++++++++
   Adw = Get_adw(1)                                    'Analogwert einlesen
   Waitms 100
   Delta = Adw - V_previous                            'Abweichung bilden
   Delta = Abs(delta)                                  'Betrag bilden
   If Delta > 4 Then                                    'Slider in Bewegung
      Busyled = 0                                      'Busy-Led ein
      Do                                               'Solange bis Slider in Ruhe
         V_previous = Adw                              'Aktuellen Wert retten
         Adw = Get_adw(1)
         Waitms 100
         Delta = Adw - V_previous
         Delta = Abs(delta)
      Loop Until Delta < 4
      Vertical = Str(adw)
      Vertical = Format(vertical , "000")
Main_loop_2:
      Display_slider
      Print "V" ; Vertical                            'Befehl an Receiver
      Status = Receive_char(receive_timeout_1)         'Wait for Receiver Busy-Char.
      If Status = Busy_char Then
         Status = Receive_char(receive_timeout_2)      'Wait for Receiver Char.
      Else


         Status = No_valid_busy_char
       End If
       Busyled = 1                                                'Busy-Led wieder aus
       If Status > 0 Then
          Row1 = "Slider ver.:"
          Display_status
          Check_communication
          Goto Main_loop_2
       End If
    End If
```

```
'+++++++++
'Tatstatur
'+++++++++
   Kbd = Getkbd()
   Kbdresult = Lookup(kbd , Kbdcode)
   Select Case Kbdresult
      Case 1 : Print "1"
               Goto Procedure_mode
      Case 2 : Print "2"
               Goto Procedure_mode
      Case 3 : Print "3"
               Goto Procedure_mode
      Case 4 : Print "4"
               Goto Procedure_mode
      Case 5 : Print "5"
               Goto Procedure_mode
      Case 6 : Print "6"
               Goto Procedure_mode

      Case 9 : Goto Display_procedures                    'Zeige die Ablauprogramme
      Case 0 : Print "0"
               Goto Relais_on
      Case 11 : Toggle Lcdbacklight                       'ON / OFF
LCD-Hintergrundbeleuchtung
                Waitms 500
                Goto Main_loop
      Case Else : Goto Main_loop
   End Select


Procedure_mode:                                           'Log-Code: P_M
   Busyled = 0
   Status = Drive mode(w(kbdresult) , S(kbdresult))
Procedure_mode_1:
   If Status = Ok Then
      Print "H" ; Horizontal                             'Befehl an Receiver
      Status = Receive_char(receive_timeout_1)            'Wait for Receiver Busy-Char.
      If Status = Busy_char Then
         Status = Receive_char(receive_timeout_2)         'Wait for Receiver Char.
      Else
         Status = No_valid_busy_char
      End If
      Busyled = 1                                         'Busy-Led wieder aus
      If Status = Ok Then
         Display_slider
      Else
         Row1 = "P_M/Command-H:"
         Display_status
         Check_communication
         Goto Procedure_mode_1
      End If
   Else
      Busyled = 1
      Row1 = "P_M/Drive_Mode:"
      Display_status
      Check_communication
      Goto Procedure_mode_1
   End If
   Goto Main_loop


Relais_on:                                                'Log-Code: Relais_ON
   Row1 = "Relais EIN..."
   Row2 = "                "
   Display_text
   Busyled = 0
   Status = Receive_char(receive_timeout_2)               'Wait for Receiver Char.
   If Status = Shooton_char Then                          'Shoot-Aktivierung empfangen
      Status = Receive_char(receive_timeout_2)            'Wait for Receiver Char.
      If Status = Shootoff_char Then                      'Shoot-Deaktivierung
empfangen
         Busyled = 1
```

```
      Display_slider
      Goto Main_loop
   End If
Else
   Busyled = 1
   Row1 = "Relais_ON:"
   Display_status
   Check_communication
   Display_slider
End If
Goto Main_loop

Display_procedures:
   Row1 = "Ablaufprogramme:"
   Row2 = "<1..6>#    <*>Abr"
   Display_text
   Wait 1
Display_procedures_1:
   Kbd = Getkbd()
   Kbdresult = Lookup(kbd , Kbdcode)
   Select Case Kbdresult
      Case 1 : Display_procedure W(kbdresult) , S(kbdresult)
               Goto Display_procedures
      Case 2 : Display_procedure W(kbdresult) , S(kbdresult)
               Goto Display_procedures
      Case 3 : Display_procedure W(kbdresult) , S(kbdresult)
               Goto Display_procedures
      Case 4 : Display_procedure W(kbdresult) , S(kbdresult)
               Goto Display_procedures
      Case 5 : Display_procedure W(kbdresult) , S(kbdresult)
               Goto Display_procedures
      Case 6 : Display_procedure W(kbdresult) , S(kbdresult)
               Goto Display_procedures
      Case 10 : Display_slider                              'Abbruch
                Goto Main_loop
      Case Else : Goto Display_procedures_1                  'Warten bis Taste gedrueckt
   End Select

End


Display_procedures:
   Row1 = "Ablaufprogramme:"
   Row2 = "<1..6>#    <*>Abr"
   Display_text
   Wait 1
Display_procedures_1:
   Kbd = Getkbd()
   Kbdresult = Lookup(kbd , Kbdcode)
   Select Case Kbdresult
      Case 1 : Display_procedure W(kbdresult) , S(kbdresult)
               Goto Display_procedures
      Case 2 : Display_procedure W(kbdresult) , S(kbdresult)
               Goto Display_procedures
      Case 3 : Display_procedure W(kbdresult) , S(kbdresult)
               Goto Display_procedures
      Case 4 : Display_procedure W(kbdresult) , S(kbdresult)
               Goto Display_procedures
      Case 5 : Display_procedure W(kbdresult) , S(kbdresult)
               Goto Display_procedures
      Case 6 : Display_procedure W(kbdresult) , S(kbdresult)
               Goto Display_procedures
      Case 10 : Display_slider                                'Abbruch
                Goto Main_loop
      Case Else : Goto Display_procedures_1                   'Warten bis Taste gedrueckt
   End Select

End
```

```
'*****************************************************************************
'* UNTERPROGRAMM: Display Status
'*
'* Uebergabe: ---
'* Ergebnis:  ---
'*****************************************************************************
Sub Display_status

Dim Status_str As String * 3

   Status_str = Str(status)
   Status_str = Format(status_str , "000")
   Row2 = "Status=" + Status_str
   Display_text
End Sub


'*****************************************************************************
'* UNTERPROGRAMM: Display_Slider
'*
'* Uebergabe: ---
'* Ergebnis:  ---
'*****************************************************************************
Sub Display_slider
   Row1 = "Schieberegler:"
   Row2 = "Hor=" + Horizontal + "  Ver=" + Vertical
   Display_text
End Sub


'*****************************************************************************
'* UNTERPROGRAMM: Display_Text
'*
'* Uebergabe: ---
'* Ergebnis:  ---
'*****************************************************************************
Sub Display_text
   Cls
   Lcd Row1
   Lowerline
   Lcd Row2

End Sub

'*****************************************************************************
'* UNTERPROGRAMM: Display_Lowerline
'*
'* Uebergabe: ---
'* Ergebnis:  ---
'*****************************************************************************
Sub Display_lowerline
   Lowerline
   Lcd Row2
End Sub
```

```
'*****************************************************************************
'* FUNKTION: Receive_Char
'*
'* Uebergabe: Timeout
'* Ergebnis:  Status or Received Character
'*****************************************************************************
Function Receive_char(byval Timeout As Integer) As Byte
   J = 0
   Do
      Serial_char = Ischarwaiting()                          'Auf Receiver-Busy
warten
      If Serial_char = True Then                            'Character empfangen
         Exit Do
      Else
         Waitms 1
         Incr J
      End If
   Loop Until J = Timeout
   If J = Timeout Then
      Receive_char = Rs232_timeout
   Else
      Inputbin Receive_char
   End If
End Function
```

```
'*****************************************************************************
'* FUNKTION: Receiver Synchronisation
'*
'* Uebergabe: ---
'* Ergebnis:  Status
'*****************************************************************************
Function Receiver_synchronisation() As Byte
   I = 0
   Printbin Synch_char                                      'Synch-Zeichen senden
   Do
      Waitms 500
      Serial_char = Ischarwaiting()                         'Auf Echo warten
      If Serial_char = True Then                           'Character empfangen
         Exit Do
      Else
         Printbin Synch_char
         Incr I
      End If
   Loop Until I = 20                                        'ca. 10 Sekunden wiederholen
   If I = 20 Then
      Receiver_synchronisation = Synch_timeout
   Else
      Inputbin Serial_char
      If Serial_char = Synch_char Then                     'Zeichen ist Synch-Zeichen
         Receiver_synchronisation = Ok
      Else
         Receiver_synchronisation = Invalid_synch_char
      End If
   End If
End Function
```

```
'*****************************************************************************
'* UNTERPROGRAMM: Check_Communication
'*
'* Uebergabe: ---
'* Ergebnis:  ---
'*****************************************************************************
Sub Check_communication
   J = 0


   Start Timer1
   Do
      Status = Receiver_synchronisation()
      Incr J
   Loop Until Status = Ok OR J = 10
   Stop Timer1
   Statusled = 1                                          'Status-LED aus
   Select Case Status
      Case Synch_timeout : Display_status
                           End
      Case Invalid_synch_char : Display_status
                                End                       'Programm stoppen
   End Select
   Statusled = 0                                          'Status-LED ein
End Sub


'*****************************************************************************
'* FUNKTION: Get_ADW
'*
'* Uebergabe: Channel
'* Ergebnis:  ADC-Wert@Channel
'*****************************************************************************
Function Get_adw(byval Channel As Byte) As Integer
   Get_adw = Getadc(channel)
   If Get_adw > 720 Then                                  'Maximalwert = 720
      Get_adw = 720
   Else
      If Get_adw < 0 Then                                 'Minimalwert = 0
         Get_adw = 0
      End If
   End If
End Function
```

```
'*****************************************************************************
'* FUNKTION: Drive_Mode
'*
'* Uebergabe: Angle, Step_Size
'* Ergebnis:  Status
'*****************************************************************************
Function Drive_mode(angle As Byte , Step_size As Byte) As Byte
   I = -1
   Angle_str = Str(angle)
   Angle_str = Format(angle_str , "00")
   Step size str = Str(step size)
   Step_size_str = Format(step_size_str , "00")
   Row1 = Angle_str + Chr(&Hdf) + " (" + Step_size_str + " Steps):"
   Display_text
   Drive_mode = Receive_char(receive_timeout_1)             'Warte auf Receiver
Busy-Char.
   If Drive_mode = Busy_char Then
      Do
         Kbd = Getkbd()                                     'Auf Abbruch pruefen
         Kbdresult = Lookup(kbd , Kbdcode)
         Select Case Kbdresult
            Case 10 : Printbin Escape_char                  'Abbruch-Taste gedrueckt
                      Row2 = "*** ABBRUCH! ***"             'Meldung ausgeben
                      Display_lowerline
                      K = 0
                      Do
                         Drive_mode = Receive_char(receive_timeout_1)
                         Incr K
                      Loop Until Drive_mode = Escape_char OR K = 20
                      If Drive_mode = Escape_char Then
                         Drive_mode = Ok
                      Else
                         Drive_mode = No_valid_escape_char
                      End If
                      Exit Do                               'Und Ausstieg
         End Select
         Drive_mode = Receive_char(receive_timeout_2)       'Warte auf Receiver Char.
         If Drive_mode < Rs232_timeout Then                 'Step-Counter empfangen
            Step_counter_str = Str(drive_mode)
            Step_counter_str = Format(step_counter_str , "00")
            Row2 = "--> " + Step_counter_str + " / " + Step_size_str + "      "
            Display_lowerline
            Drive_mode = Receive_char(receive_timeout_2)    'Wait for Receiver Char.
            If Drive_mode = Shooton_char Then               'Shoot-Aktivierung empfangen
               Row2 = "--> " + Step_counter_str + " / "
               Row2 = Row2 + Step_size_str + "  " + Chr(&Hff) + Chr(&Hff) + Chr(&Hff)
               Display_lowerline
               Drive_mode = Receive_char(receive_timeout_2) 'Wait for Receiver Char.


               If Drive_mode = Shootoff_char Then           'Shoot-Deaktivierung
empfangen
                  Row2 = "--> " + Step_counter_str + " / " + Step_size_str + "      "
                  Display_lowerline
                  Incr I
               Else
                  Drive_mode = No_valid_shootoff_char
                  Exit Do
               End If
            Else
               Drive_mode = No_valid_shooton_char
               Exit Do
            End If
         Else
            Exit Do
         End If
      Loop Until I = Step_size
      If I = Step_size Then
         Drive_mode = Ok
      End If
   Else
      Drive_mode = No_valid_busy_char
   End If
End Function
```

```
'*****************************************************************************************
'* UNTERPROGRAMM: Display_Procedure
'*
'* Uebergabe: Angle, Step_Size
'* Ergebnis:  ---
'*****************************************************************************************
Sub Display_procedure(angle As Byte , Step_size As Byte)
   Busyled = 0
   Angle_str = Str(angle)
   Angle_str = Format(angle_str , "00")
   Step_size_str = Str(step_size)
   Step_size_str = Format(step_size_str , "00")
   Row1 = "Programm #" + Str(kbdresult) + ":"
   Row2 = Angle_str + Chr(&Hdf) + " (" + Step_size_str + " Steps)"
   Display_text
   Wait 3
   Busyled = 1
End Sub


'*****************************************************************************************
'* ISR: ISR_Timer1
'*
'* Uebergabe: ---
'*****************************************************************************************
Isr_timer1:
   Load Timer1 , Timer1_reload
   Toggle Statusled
Return


'*****************************************************************************************
'*  ZUORDNUNG DER TASTEN
'*****************************************************************************************
' 255 --> undefiniert oder keine Taste gedrueckt
' *   --> 10
' #   --> 12

Kbdcode:
Data 1 , 2 , 3 , 255 , 4 , 5 , 6 , 255 , 7 , 8 , 9 , 255 , 10 , 0 , 11 , 255
```

Eigenständigkeits - Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe.

Stellen, die wörtlich oder sinngemäß aus Quellen entnommen wurden, sind als solche kenntlich gemacht.

Diese Arbeit wurde in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt.

Oetz, im Oktober 2011

Christoph Köhle

Danke

Ich möchte mich an dieser Stelle bei allen bedanken, die mich bei dieser Arbeit und während meines Studiums begleitet und unterstützt haben. Diese Arbeit wäre ohne Unterstützung wohl kaum in diesem Umfang entstanden.

An erster Stelle gilt der Dank meiner Familie. Meiner Frau Barbara und meinen Kindern, Kathrin und Elias, die immer Verständnis für mich hatten, wenn ich weder Zeit noch Geduld für sie hatte.

Mein Dank gilt im Besonderen Herrn Prof. Dr.-Ing. Thomas Beierlein für die fachliche und organisatorische Betreuung bei dieser Diplomarbeit und während des vorangegangenen Studiums.

Großer Dank gilt ebenfalls meinem betrieblichen Betreuer und Initiator dieser Arbeit Herrn Dipl.-Ing. (FH) Christian Neuner, der nicht nur für technische Auskünfte jederzeit für mich da war, sondern auch eine moralische Stütze während meiner ganzen Studienzeit.

Auch danke ich meinen Vorgesetzten Herrn Direktor HR Mag. Dr. Elmar Märk und Herrn Abteilungsvorstand RR Dipl.-Ing. Hans Hudovernik dafür, dass sie es mir ermöglicht haben an den Vorlesungen und Prüfungswochen teilzunehmen.